AS
ASCII Learning System

アスキー・ラーニングシステム　②実用コース

実用
MS-DOS

村瀬康治 著

改訂新版

アスキー出版局

アスキー・ラーニングシステム
実用 MS-DOS
改訂新版
村瀬康治 著
②実用コース
ASCII LEARNING SYSTEM

アスキー・ラーニングシステム　②実用コース

実用 MS-DOS

村瀬康治 著

改訂新版

アスキー出版局

MS-DOS Learning System 全3巻の構成

アスキー・ラーニングシステム MS-DOS 編は、『入門 MS-DOS』『実用 MS-DOS』『応用 MS-DOS』の3部から構成されています。

入門編および実用編では、ビジネスソフトのユーザーをはじめとする、広く一般的なユーザーを対象に、MS-DOS の基礎知識とその活用法を初歩からわかりやすく解説します。また応用編では、MS-DOS 上で各種のソフトウェアを開発するプログラマー、あるいは MS-DOS の内部を詳しく知りたいユーザーを対象に、MS-DOS システムの初歩から、アセンブラや C 言語による実際のプログラミング例を豊富にまじえながら、高度な知識に至るまでを解説します。

それぞれの巻は次のように構成されています。

入門 MS-DOS：MS-DOS 上の各種のビジネスソフトなどを利用しようとする人を対象に、MS-DOS の基本的な機能や操作法をやさしく、ていねいに解説します。とくに、MS-DOS によって初めて OS に接する人も、楽に読み進めるよう十分配慮されています。この入門編により、各種のソフトを活用するための MS-DOS の知識が、バランスよく得られるようにまとめてあります。

実用 MS-DOS：MS-DOS の基礎知識を学んだ人に対して、MS-DOS マシン本来の実力をフルに発揮させ、各種のソフトを効果的に使いこなすための知識とノウハウを提供します。ATOK、松茸、VJE などによる日本語入力システムの使い方や漢字コードの知識、ハードディスクの各形態のフォーマット処理や初期設定、ハードディスクの効果的運用法、RAM ディスクやディスクキャッシュ、それに EMS を実現する具体例とその利用法、さらに各種エディタのやさしい使い方などを実例解説します。また、目的別による MS-DOS の主要コマンドの使い方を、コマンドリファレンスの形式と実行例により示します。この実用編は、ビジネスソフトのユーザーにも、ソフトウェア開発者にも、MS-DOS マシンを使いこなす重要な知識を提供するでしょう。

応用 MS-DOS：MS-DOS 上でソフトウェアを開発する人や、MS-DOS の内部構造を詳しく知りたい人に対して、豊富な図や実際のプログラムを示しながら、MS-DOS システムの構造をその基本から細部まで、ていねいに解説します。また、システムコールの解説や、アセンブラによるソフトウェア開発、さらに C 言語の開発環境について解説し、そのプログラミングを実習します。この応用編は、これから MS-DOS システムを学ぶプログラマーにとって、たいへん親切に書かれた最適の参考書となるでしょう。

このシリーズは、MS-DOS のバージョン 3.3 をもとに書かれていますが、それ以前のバージョンをお使いの方も、まったく支障なく利用できます。この MS-DOS 3部作を読むにあたり、MS-DOS マシンを操作できれば理想的ですが、いずれも豊富な実行例をもとに解説していますので、紙上体験だけでも大きな効果が得られるでしょう。

はじめに

MS-DOSマシンの環境が成熟してきました。アプリケーションソフトの質も、マシンの性能や各種の周辺装置も格段に強力になりました。今日、仕事に使うマシンの装備といえば、本体メモリは640Kバイトのフル実装、さらに2MバイトのRAMボードを増設してディスクキャッシュやRAMディスクを実現し、そして40Mバイト程度のハードディスクを付ける——この程度がもはや〝常識的〟となりました。

MS-DOSマシンの環境は、1988年あたりを境に急激に進展しましたが、これは各ソフトウェアメーカーの技術力が向上し、MS-DOS本来の能力を十分に活用した優れたアプリケーションソフトが開発できるようになったこと、また、パーソナルコンピュータ全体の性能が向上し、さらにメモリやハードディスクの価格が劇的に低下したことなどによる相乗効果でしょう。もちろん私たちユーザーの、パーソナルコンピュータの活用が高度になり、より快適な環境を求める声が強くなってきたことも大きな要因です。

本書は、このような成熟したMS-DOSの環境に対応するために、MS-DOSマシンをより快適に活用するためのノウハウを提供します。現在の代表的な日本語入力システム(FEP)の利用法、ハードディスクの初期設定や拡張フォーマットによるパーティション分割などを実例解説し、さらに実際にRAMディスク／ディスクキャッシュ／EMSなどを実現した上でその運用法などを解説します。また、MS-DOSの主要コマンドの使い方を用途別のコマンドリファレンスとして活用できるようにまとめてあります。

さて、本書を含むMS-DOSシリーズ3部作は、初版より5年目を迎え、その発行部数(3部合計)は50万部を超えました。たいへんうれしく思う反面、一層大きな「責任」を感じています。そこでこれを機会に、MS-DOSやMS-DOSマシンの進歩、アプリケーションソフトの発展などの環境の変化に合せて、3部作を全面的に大改訂いたしました。本書も、現在の環境の基本部分を十分にカバーできたものと思います。

MS-DOSの環境は今後とも発展していき、次世代のOS/2時代を迎えても、かなり長い期間にわたり並行して使われていくでしょう。これからの私たちには、マシンの装備を適切に行うとともに、よいアプリケーションソフトを選び、それらを最大限に活用するための知識が一層強く求められます。そのために本書が活用され、よりよい環境のもとで、少しでも快適に仕事が行えることを願っています。

1989年10月　　村瀬康治

目　次

3章　RAM ディスク、ディスクキャッシュ、EMS

4章　ハードディスクの使い方

5章 エディタによる簡単な文書ファイルの作成および編集

6章 MS-DOS主要コマンド目的別解説

APPENDIX

イラスト:長谷川 弘

1章
日本語入力機能

私たちが使うコンピュータは、漢字やひらがななどの日本語が自由に使えなければなりません。今日では、日本語が使えるコンピュータはあたりまえとなりましたが、MS-DOS が誕生した 1981 年当時は、大型マシンを含めて、コンピュータで漢字を扱うことはそう簡単ではありませんでした。漢字の入力方法も、現在のような能率的な「かな－漢字変換」や「文節変換」が十分に発達しておらず、漢字の文字盤(漢字タブレット)から目的の字を 1 文字ずつ拾い出しては入力するようなことも行われていました。

その後、日本語入力の機能は、パーソナルコンピュータやワープロ専用機の普及とともに急速に発展することになりますが、その発展に、MS-DOS は大きな役割を果しています。巧妙な「シフト JIS コード」(2 章で解説)を考案・採用し、MS-DOS の「デバイスドライバ」の機能 ── 外部のプログラムを、あたかも MS-DOS 自身のプログラムであるかのように、MS-DOS に組み込む機能(後述)── をうまく利用することによって、日本語入力機能は飛躍的に発展しました。

本章では、その現在の代表的な日本語入力システム、「ATOK(エートック)」、「松茸(まつたけ)」、「VJE(ブイジェイイー)」、について紹介しましょう。なお、ここでの実行例は、もっとも一般的な機種である PC-9801 シリーズ上での例を示しています。

1.1　日本語入力システム「FEP」(フロントエンドプロセッサ)

日本語入力システムは、いつの頃からか「フロントエンドプロセッサ」——略して「FEP」と呼ばれるようになりました。「フロントエンド」という言葉は、日本語入力システムが、MS-DOSの「デバイスドライバ」の形式をとり、アプリケーションプログラムから独立したシステムであることを言い表したものでしょう。

日本語入力システムに「FEP」という言葉を使ったのは、1984年に発売された日本語入力システム「VJE」の製品名を「日本語入力フロント・プロセッサ」と称したのが最初であると言われています。

現在の代表的な日本語入力システムには、ワープロ「一太郎」に使われている**ATOK**(エートック)や、同じく「新松」に使われている**松茸**、それに、当初から単独の日本語入力システムとして作られた**VJE**などがあげられますが、これらはいずれも、MS-DOSのデバイスドライバとして動作する、いわゆる「FEP」です。現在使われているほとんどの日本語入力システムは、デバイスドライバの形態をとっているため、それぞれ特定のアプリケーションプログラム専用のシステムとして使われるだけでなく、ほかのさまざまなアプリケーションプログラムにも自由に利用することが可能なのです。逆に言えば、そのようなことができる日本語入力システムが「FEP」であるわけです。

■ ワープロは、「FEP＋各種編集機能」

一太郎や新松などに代表される、現在のいわゆる「ワープロソフト」のプログラムの内部構成は、大きく分けて2つの部分——「日本語入力システム」と、「各種編集機能や文書ファイル管理機能」から成り立っています。つまり、「FEPの部分」と「それ以外の部分」です。この2つの部分は互いに独立したプログラムであり、独立した動作をしているのですが、ワープロを操作しているユーザーが、別々のプログラムを実行しているという感じを受けることはないでしょう。これら2つの機能の間に、操作上の明確な〝境目〟はなく、互いの機能のつながりがスムーズであるため、全体が1つでできたプログラムであるかのように思えるのです。

「FEPの部分」というのは、本章で解説する日本語入力システムの部分であり、文字のキー入力と、それを「かなー漢字変換」や「文節変換」する部分です。「辞書ファイル」はこのFEPに属します。「FEPの部分」に対して「それ以外の部分」の主な機能は、各種編集機能や、文書ファイルを管理する機能、それに印刷機能と言ったものと考えればよいでしょう。

たとえば、ワープロ「一太郎」の日本語入力システムには、「ATOK」と呼ばれるFEPが使われています。その「FEPの部分」と「それ以外の部分」とは互いに独立していますので、一太郎から「FEPの部分」(つまりATOK)だけを単独に取り出せば、別のアプリケーションプログラムの日本語入力にも利用できるはずです。現実に「ATOK」は、一太郎専用の(一太郎だけにしか使えない)FEPとしてではなく、各種のアプリケーションプログラムからも、「一太郎と同じ操作で使える」(当然ですが)日

本語入力システムとして利用されています。このことは、「新松」のFEPである「松茸」についてもまったく同じです。

このように、各種の日本語入力システムは、「FEP」(つまりデバイスドライバ)という形態をとったことにより、さまざまなアプリケーションプログラムから共通に利用することができるようになりました。そのことによって利用の幅が格段に広がったとともに、日本語入力システムは、より優れた製品、より快適に使える製品へと急速に発展してきたのです。

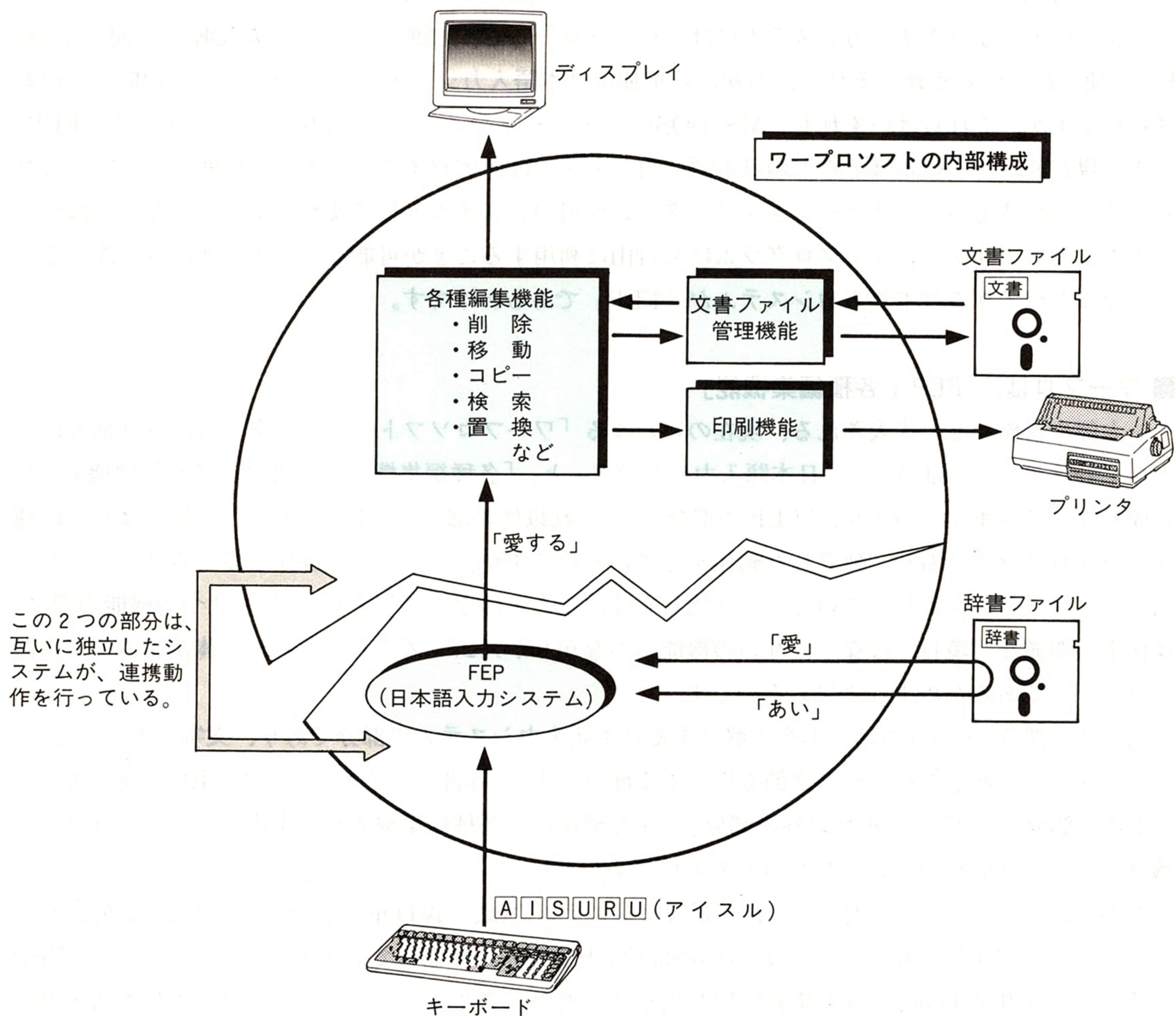



図 1.1　ワープロソフトの内部構成

■FEP はデバイスドライバ

デバイスドライバの仕組みを、ここでは日本語入力システムに関連する範囲で、簡単に説明しておきましょう。

MS-DOS は、パーソナルコンピュータが動作するために必要な、もっとも基本的な機能を提供する「基本ソフトウェア」ですので、それほど多くの機能が備っているわけではありません。ユーザーが必要とする機能が MS-DOS に備っていなければ、ユーザー側(ソフトハウスや個人のプログラマ)がその機能を実現するための独自のプログラムを作成しなければなりません。日本語入力機能も、MS-DOS に備っていない機能の1つであり、ユーザー側が独自のプログラムによって実現しなければなりません。

ユーザー側が作成するプログラムは、通常、MS-DOS の「外部プログラム」として実行される形態をとります(MS-DOS の外部コマンドや各種のビジネスソフトなどで、「.COM」や「.EXE」のファイルタイプ(拡張子)が付いたプログラムファイルは、すべてこの形態です)。これらのプログラムは、普通、ある特定の目的のために動作するものであり、関係のない別のプログラムから利用することはできません。しかし、作成するプログラムの形態を「外部プログラム」ではなく、「デバイスドライバ」として作成すれば、そのプログラムは、あたかも MS-DOS に備っている機能の1つであるかのように動作します。つまり、「MS-DOS の機能」と同等に取り扱うことができるため、さまざまなアプリケーションプログラムから共通に利用することが可能になるのです。

結局、「デバイスドライバ」というプログラムの形態は、ユーザー側が作成するさまざまな機能のプログラムを、「MS-DOS 自身の機能であるかのように見せかけるためのプログラム形式」と考えればよいでしょう。

ではここで、日本語入力システムが「MS-DOS 自身の機能」として働いていることを、ワープロソフト(ここでの実行例は一太郎)を例に検証してみましょう(あまりに簡単な方法で恐縮ですが)。

まず、起動している一太郎を「終了」して MS-DOS におりた状態(プロンプト「A>」が表示された状態)を考えます。「終了」した以上、コンピュータのメモリ上の一太郎のプログラムは消滅しているはずです。ところがそのような状態でも、一太郎の日本語入力の機能は生きているのです。

CTRL ＋ XFER を入力する(FEP ON)

①一太郎を終了する

〈一太郎〉

Q・終了

半角英数カナ

②FEPがOFFのMS-DOS

A＞

③日本語入力システムが機能する
FEPがONの状態

A＞

生きているFEP　連ローマ字漢字

ATOKの日本語
入力ライン

④文節変換などを使って、
自由に日本語入力ができる

A＞生きているFEP

それは、遥かなる　連ローマ字漢字

⑤FEPがOFFのMS-DOS

A＞DIR

再び CTRL ＋ XFER を入力する(FEP OFF)

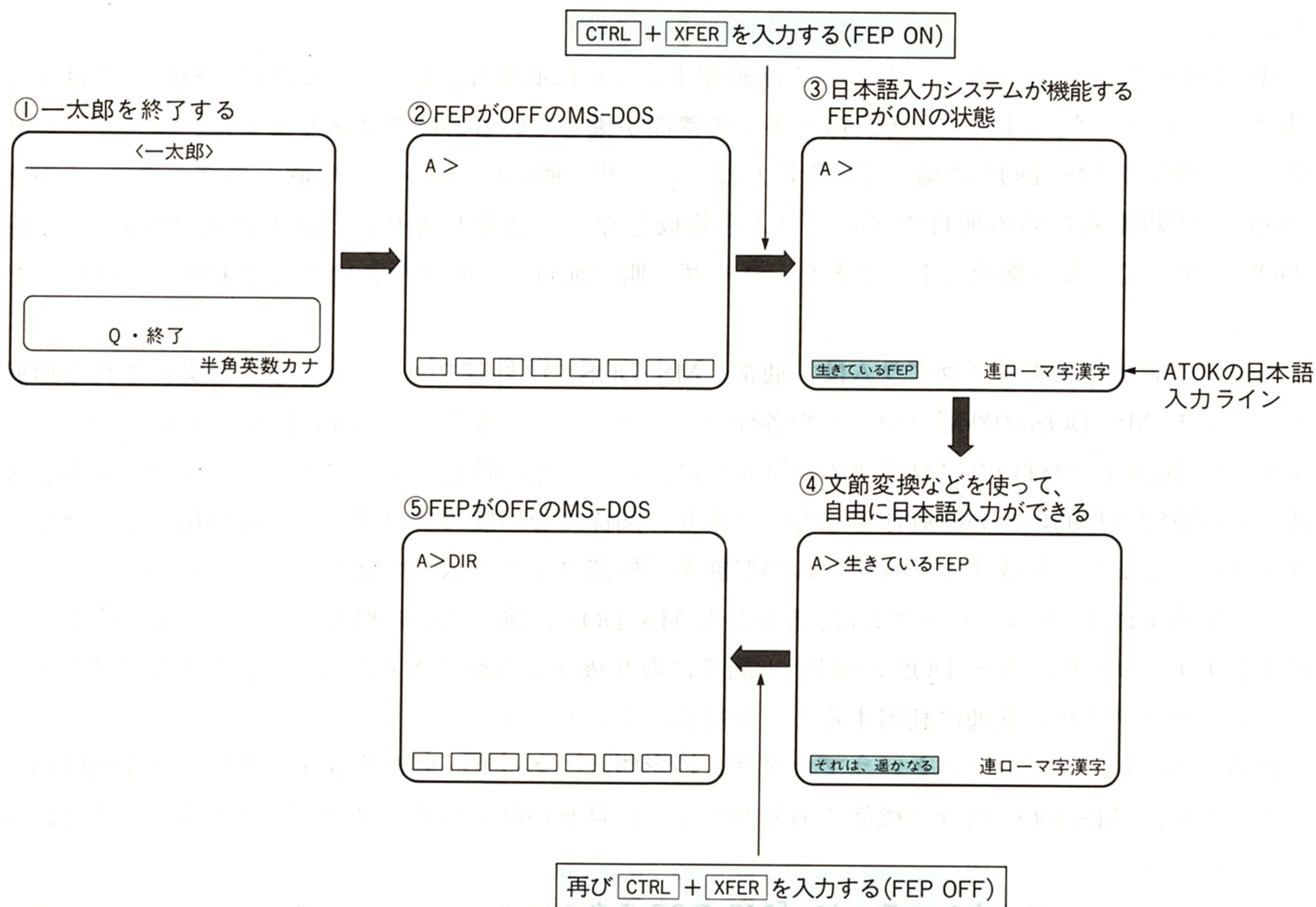



図 1.2　一太郎を「終了」した後、プロンプト「A＞」に続けて日本語を入力する

一太郎を「終了」して、MS-DOSのコマンドレベルの状態になっても、このように日本語の入力ができることから、一太郎自身のプログラムは消滅しても、一太郎で使っていた日本語入力システム(ATOK)は生き残っていることがわかります。つまり、ATOKは「MS-DOSとともに存在している」と考えることができます。「MS-DOSに組み込まれている」と言ってもよいでしょう。ですからこのATOKは、表計算ソフトのMultiplanや、データベースソフトのdBASE III、それにエディタのMIFESなど、各種のアプリケーションプログラムで共通に利用することができるのです。

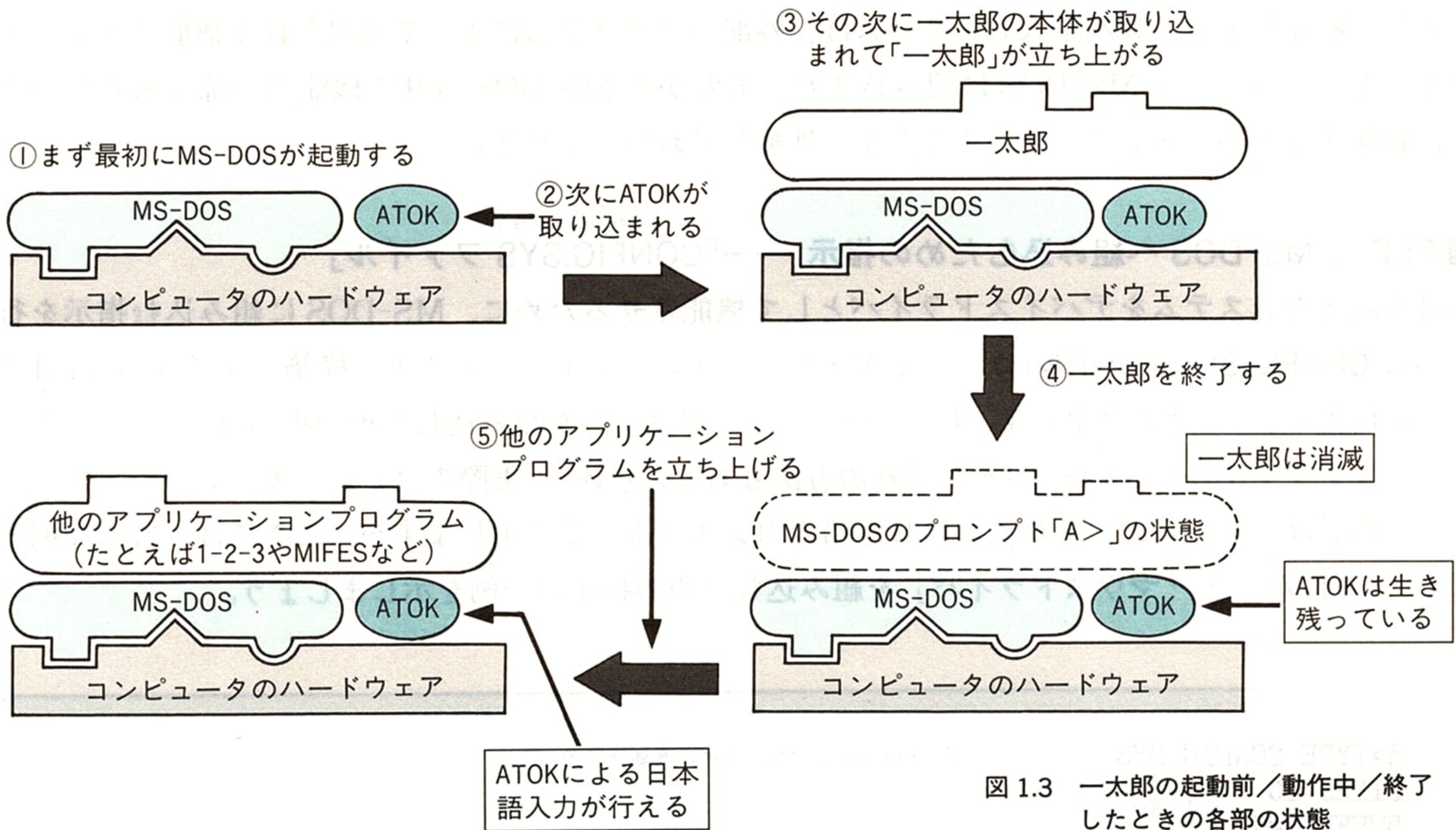



図 1.3　一太郎の起動前／動作中／終了したときの各部の状態

私たちにおなじみのデバイスドライバには、このようなFEPのほか、マウスの基本動作プログラムである「マウスドライバ」があります。この場合も、同一のマウスドライバが、「CANDY3」にも「Z' Staff」にも「Multiplan」にも、各種のアプリケーションプログラムで共通に利用されるのです（ただし、アプリケーションプログラムによっては、付属のマウスドライバのみ使用可能な場合もある）。

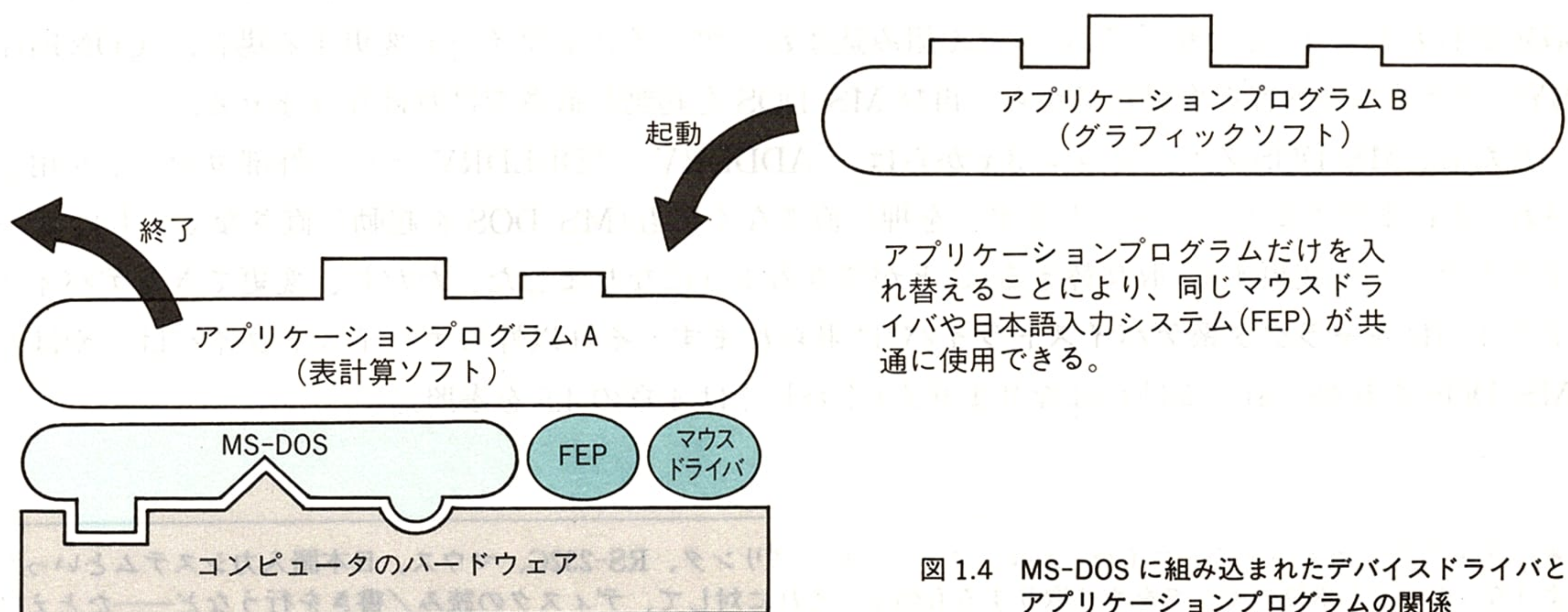



図 1.4　MS-DOS に組み込まれたデバイスドライバとアプリケーションプログラムの関係

これまで見てきたように、FEPと呼ばれている日本語入力システムは、MS-DOSのデバイスドライバとして動作します。日本語入力システムは、外部のプログラムによって実現される機能ですが、デバイスドライバとしてMS-DOSに組み込まれ、あたかもMS-DOS自身の機能の一部であるかのように動作するものであることを、ここでよく理解しておいてください。

■ FEPをMS-DOSへ組み込むための指示 ──「CONFIG.SYS(コンフィギュレーション)ファイル」

日本語入力システムをデバイスドライバとして機能させるために、MS-DOSに組み込む指示を行うのは**CONFIG.SYSファイル**(コンフィギュレーションファイル：システム構築ファイル)の役目です。日本語入力システムに限らず、すべてのデバイスドライバを組み込むための指示は、コンフィギュレーションファイルで行います(それ以外の方法もある。後述)。実際のコンフィギュレーションファイルの内容は、このあと、それぞれの日本語入力システムの節で示しますが、ここでは一般的な例として、「ATOK」と「マウスドライバ」を組み込むための指示の一例を示しましょう。

```
A>TYPE CONFIG.SYS⏎      ……CONFIG.SYSファイルを表示する
FILES=10       } 3.1章を参照
BUFFERS=20     }
DEVICE=MOUSE.SYS        ……マウスドライバを組み込む
DEVICE=ATOK6A.SYS   } ATOK6(日本語入力システム)を組み込む
DEVICE=ATOK6B.SYS   }

A>
```

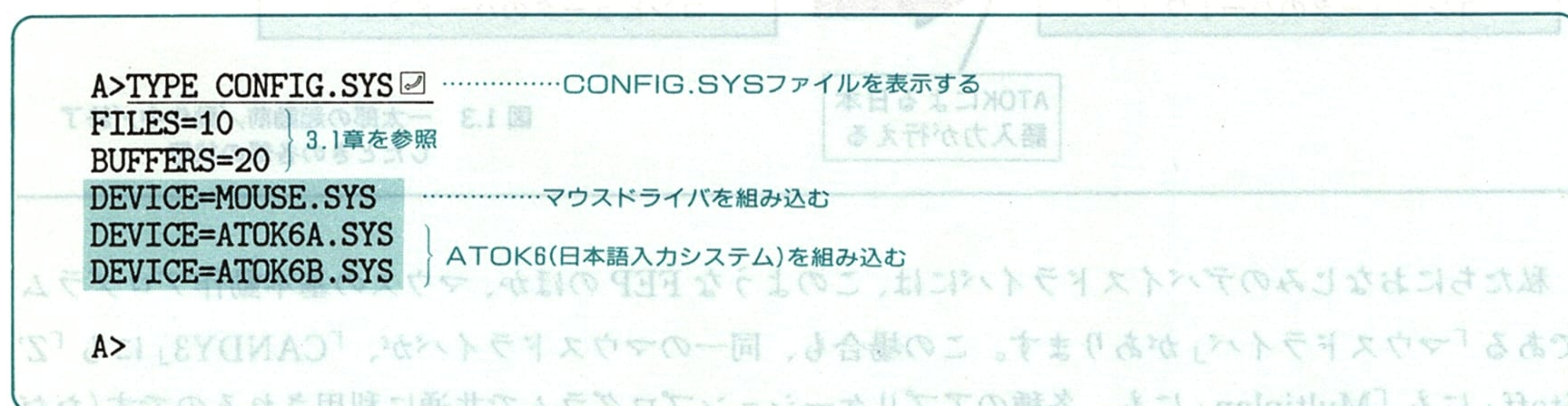



図 1.5　CONFIG.SYSファイル内のデバイスドライバを組み込むための指示

CONFIG.SYSファイルの内容は、MS-DOSの起動時に読み出され、その内容にしたがった各種の設定が行われます。したがって、いったん組み込まれたデバイスドライバを変更する場合は、CONFIG.SYSファイルの内容を変更してから、再びMS-DOSを起動し直さなければなりません。

ただし、MS-DOSのバージョン3.*x*からは、「ADDDRV」、「DELDRV」という外部コマンドが用意され、それまでのようにリセットボタンを押し直さなくても(MS-DOSを起動し直さなくても)、デバイスドライバを変更する(取り替える)ことができるようになりました。ただし、変更できるデバイスドライバは**キャラクタ系デバイスドライバ***に限られます。それ以外のデバイスドライバは、やはりMS-DOSを起動し直さなければなりません(くわしくは4章の4.5を参照)。

---

* 「キャラクタ系デバイスドライバ」とは、コンソール、プリンタ、RS-232C、マウス、日本語入力システムといったような、1バイト単位のデータをやり取りするもので、それに対して、ディスクの読み／書きを行うなど──たとえばRAMディスクのデバイスドライバなど──のデバイスドライバは、「ブロック系デバイスドライバ」と呼ばれる

次に、FEP の基本的な動作と操作法について解説しておきましょう。

■ FEP の基本操作

FEP の基本操作は、次の図の①②および③に要約されます。

A＞にちようびのごご、

A＞日曜日の午後、

①ひらがなの読みを入力

ローマ字あるいはカタカナキーで入力する

②かなー漢字文節変換

スペースバーあるいは XFER などの変換キーで行う

変換された漢字まじり文

辞書ファイルが参照される

③再変換／修正

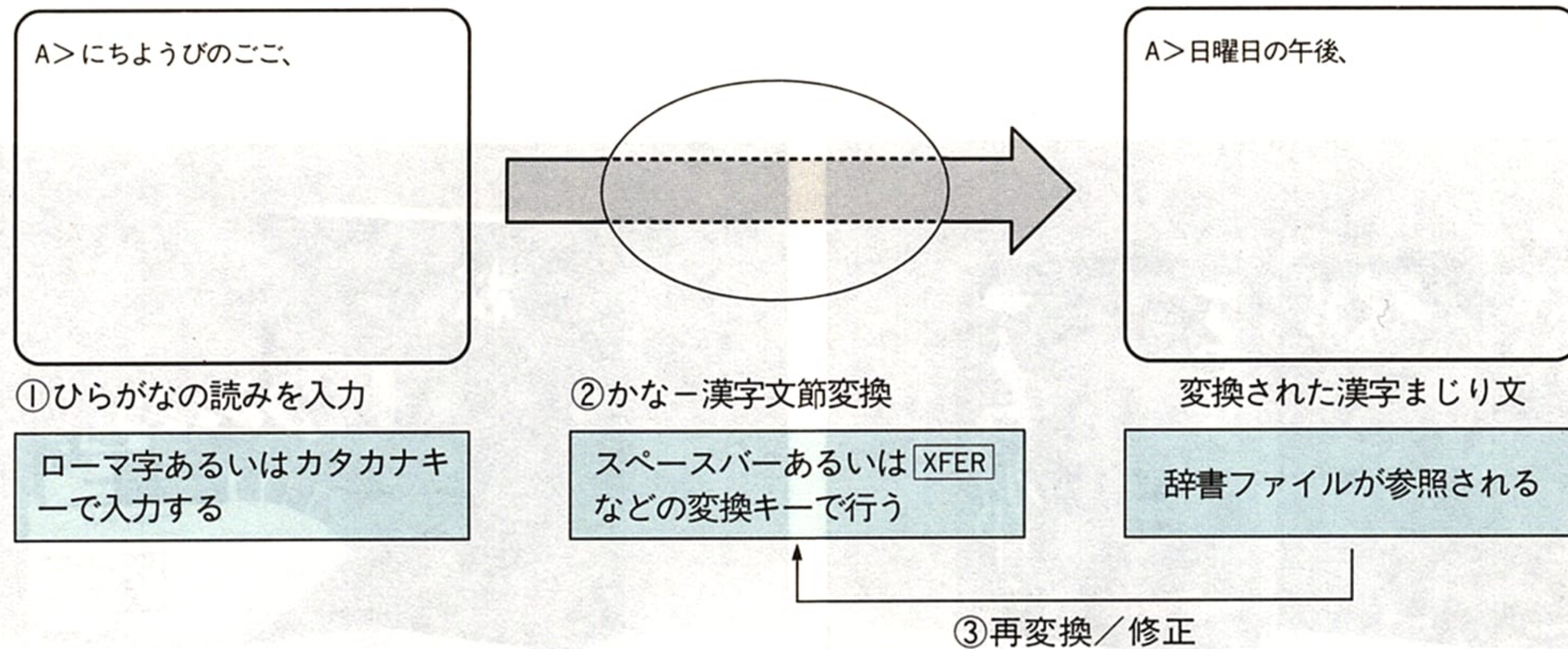



図 1.6 FEP の基本的な操作

[①の部分]

ローマ字、あるいはカタカナキーを使って、文の「読み」を入力します。ローマ字で入力しても、カタカナキーで入力しても、入力されたものはどちらも「ひらがな」に自動変換されて画面上に表示されます。

[②の部分]

次に、入力したひらがなの「読み」に対して、「かなー漢字変換」とか「文節変換」の操作を行い、目的とする漢字まじり文に変換します。この変換のためのキーは、それぞれの FEP によって異なりますが、一般的には、スペースバーや XFER を押すことによって変換が行われます。

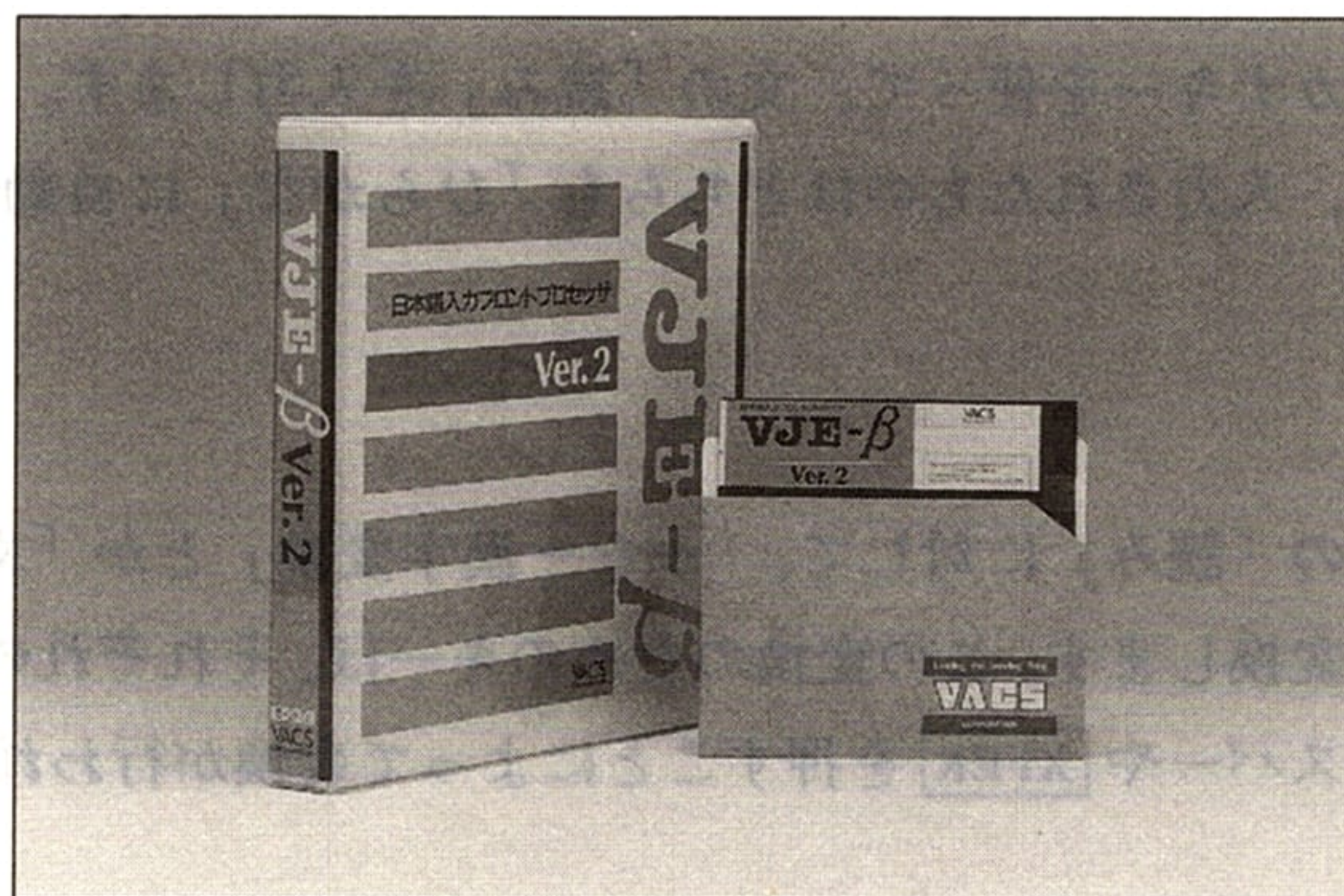
[③の部分]

目的の語に変換されなかった場合、その修正のための再変換などを行います。

一般的なFEPの基本操作は、この①②③の操作を繰り返すことによって、目的の文字列を入力していきます。①の操作のとき、複数の文節の「読み」を入力して、それを②で一度に変換するやり方が、現在もっとも一般的に使われている「連文節変換」と呼ばれる変換形式です。上の例の「にちようびのごご、」は、「にちようびの／ごご、」という、2つの文節からなる連文節です。

さて以上のようなことが、個々の日本語入力システムを解説するにあたっての予備知識です。では次節から、代表的ないくつかのFEPを取り上げ、そのファイル構成や、基本的な使い方を紹介していきましょう。

一太郎バージョン3とバージョン4
（ジャストシステム社）

新松（管理工学研究所）

VJE-β（バックス社）

1.2　一太郎に付属の日本語入力システム「ATOK(エートック)」

まず、現在もっともシェアの高いワープロソフト、「一太郎」(ジャストシステム社)の日本語入力システム(略して FEP とも記します)である「ATOK」から紹介しましょう。一太郎の歴史は、1983 年にアスキーから発売された「JS-WORD」というワープロソフトに始まります。JS-WORD の FEP には「KTIS(ケーティス)」という名前が付けられていましたが、それらが一太郎や ATOK の前身となりました。

そして 1985 年には「jxWORD 太郎」へと発展し、その FEP は「ATOK」(ATOK3)と名付けられました。ATOK という名前の由来は、「Automatic Transfer Of Kana-kanji」からとられたと伝えられています。「jxWORD 太郎」はその後まもなく「一太郎」(FEP は ATOK4)に発展して「一太郎－ATOK」路線が定着しました。その後は数回のバージョンアップが繰り返されて今日に至っています。とくに 1989 年には、主に動作環境に関する大幅な変更が行われた「バージョン 4」(ATOK7 が付属)が発売されています。

ATOK は、ワープロソフト「一太郎」の FEP として、そのシステムディスクに組み込まれているものであり、ATOK 単独では販売されていません。しかし一太郎のシステムディスクから ATOK の部分を取り出して、ユーザーの任意のディスクに組み込むことにより、ほかのいろいろなアプリケーションプログラム上から、ATOK の日本語入力機能を利用することができるのです。

■ ATOK

ここでは PC-9800 シリーズ用の ATOK6 を中心に紹介しましょう。ATOK6 は、上で述べたように、ワープロ「一太郎」に使われている FEP です。一太郎のバージョンアップとともに、ATOK のバージョンも、ATOK4、5、6、7 とアップされてきましたが、このいずれも、基本的な機能や操作法はほとんど同じです。本書では、それらの代表として、ATOK6 を取り上げますが、本書で解説する範囲のことは、どのバージョンの ATOK にもほぼ当てはまります(ATOK7 に関しては、ファンクションキーの割り当てなどが多少異なる部分もある)。

ATOK を、私たちが使っている任意のアプリケーションプログラムで利用するには、一太郎の各種のプログラムファイルから、FEP である「ATOK」に関する部分を取り出し、それぞれのアプリケーションプログラムのディスクに組み込みます。この FEP の移植により、たいていのアプリケーションプログラムから、ATOK の日本語入力機能を利用することができます。このことを図解してみましょう。

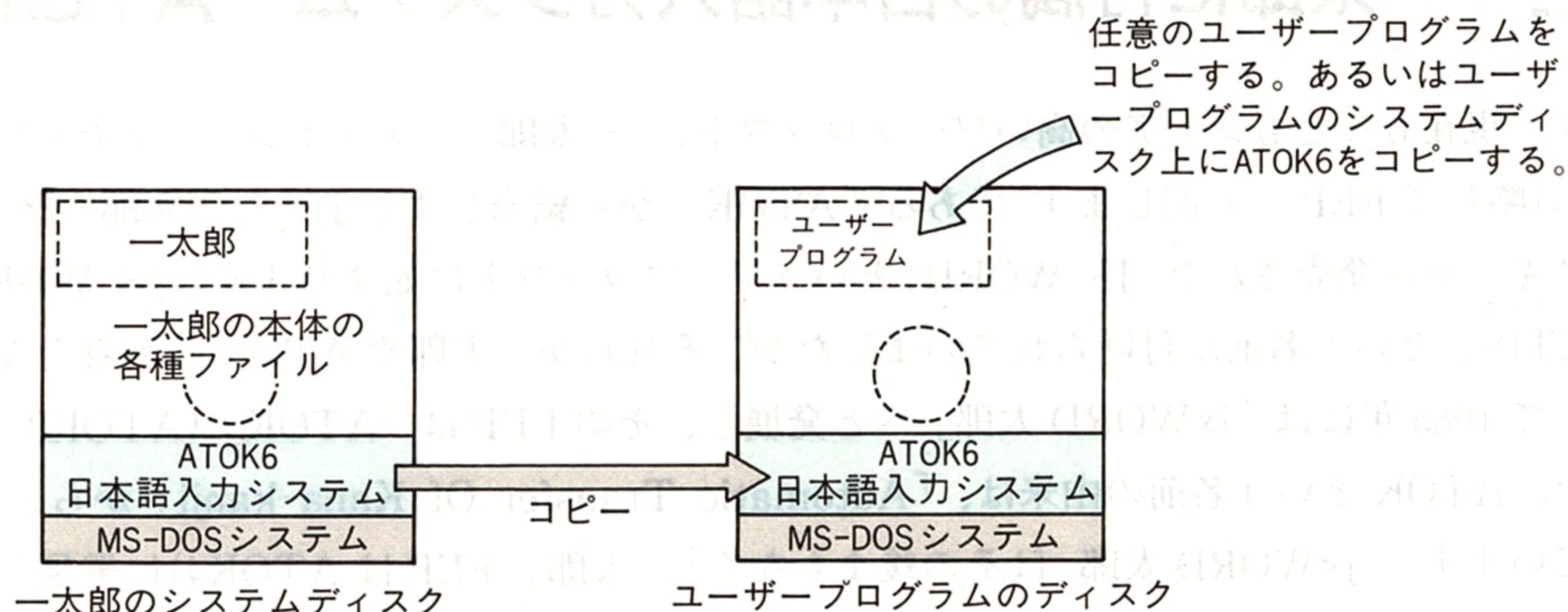



図 1.7　一太郎のディスクに含まれる日本語入力システム「ATOK」とその移植

前節で述べたように、ATOKをはじめとするFEPは、デバイスドライバとして、ワープロソフトの本体から独立していますので、この図のように、ATOKに関する部分だけを自由に取り出し、任意のアプリケーションプログラムのディスク上に組み込むことが可能なのです。本章では、FEPの機能(日本語入力機能)の紹介が目的ですので、一太郎の本体には言及せず、一太郎のシステムディスクからATOKのみを取り出して、ATOKが機能する新しいMS-DOSシステムディスクを作成し、それをもとに実習解説を行いましょう。

■ ATOKを組み込んだシステムディスクの作成

ユーザーが使用する各種のアプリケーションプログラム上でATOK6のFEPを利用できるように、一太郎からATOKを移植する実例を示しましょう。さきにも述べたように、ここでは一太郎Ver.3のATOK6を例にしますが、ほかのバージョンのATOKも、本書で解説するような基本的な範囲ではほとんど同じです。また、ATOK6が動作するためには、ATOK6のみで約100Kバイトのメインメモリを占有しますので、考慮しておいてください。

ではまず、一太郎Ver.3のシステムディスクに含まれている全ファイルのなかから、ATOK6(つまりFEP)に関連するファイルを示します。

```
A>DIR        ……一太郎のシステムディスクに含まれるファイルの内容

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

COMMAND  COM    17190  83-09-19   16:20
ATOK     DIC   454144  87-07-24   13:37  ……辞書ファイル
JXW      EXE   514264  87-06-15   12:00
JXW      OVL    78416  87-06-15   12:00
JXW      COM      942  87-06-15   12:00
CACHE    SYS     6918  87-06-15   12:00
ATOK6A   SYS    54265  87-06-15   12:00 } ATOK6本体のプログラム。A、Bの2つに分かれている
ATOK6B   SYS    21372  87-06-15   12:00 }
GAIJ             9392  87-06-15   12:00  ……外字ファイル。外字を使わない場合は必要ない
FORMAT   COM    14742  85-05-19   13:59
PSET     PRD     8294  87-06-15   12:00
CONFIG   SYS       89  87-06-15   12:00  ……ATOK6をMS-DOSに組み込むための
                                           CONFIG.SYSファイル
AUTOEXEC BAT       35  87-06-15   12:00
JXWSYS           1550  89-08-21   17:02
       14 個のファイルがあります.
     2048 バイトが使用可能です.

A>
```

図 1.8　一太郎のシステムディスクに含まれる ATOK6 に関する各種のファイル

このリスト上の色でマークされたいくつかのファイルが、ATOK6 に関するものです。これらのファイルを日常使用する MS-DOS システムディスクや、各種のアプリケーションプログラムのディスク上にコピーすればよいのです。そのことを次ページの**図 1.9** で示しましょう。

では，この作業の実行例を示しましょう。まず、図 1.9 の例 1(上側)に示されている手順により、フォーマット処理した空ディスクに MS-DOS システムをコピーし、その上に ATOK6 をコピーして、ATOK6 を組み込んだ新しい MS-DOS システムディスクを作成します。

MS-DOS システム(COMMAND.COM を含む)をコピーするには、

A>FORMAT　B：/S	あるいは	A>SYS　B：

の 2 つのコマンドが用意されていますが、ここでは FORMAT コマンドを使った実行例を示しましょう。コピー元にする MS-DOS システムには、MS-DOS システムディスクや、一太郎の常用ディスク(日常の作業に使っているディスク)に含まれる MS-DOS システムが使えます。

まず、コピーする MS-DOS システムのマザー(コピー元)となる MS-DOS システムディスクや、一太郎の常用ディスクに、FORMAT プログラムの「FORMAT.EXE(あるいは.COM)」が含まれてい

るかどうかを確認してください。もし含まれていない場合には、あらかじめFORMAT.EXE(.COM)をコピーしておいてください(あるいは、そのプログラムが存在する別のディスクを使ってFORMATプログラムを起動した後、本来のディスクに入れ替えてもよい)。

では、MS-DOSシステムディスク、または一太郎の常用ディスクをドライブA：にセットし、コピー先のディスクをドライブB：にセットします。そこで「/S」付きのFORMATコマンドを実行し、ドライブB：のディスクをフォーマットするとともに、ドライブA：のディスク上のMS-DOSシス

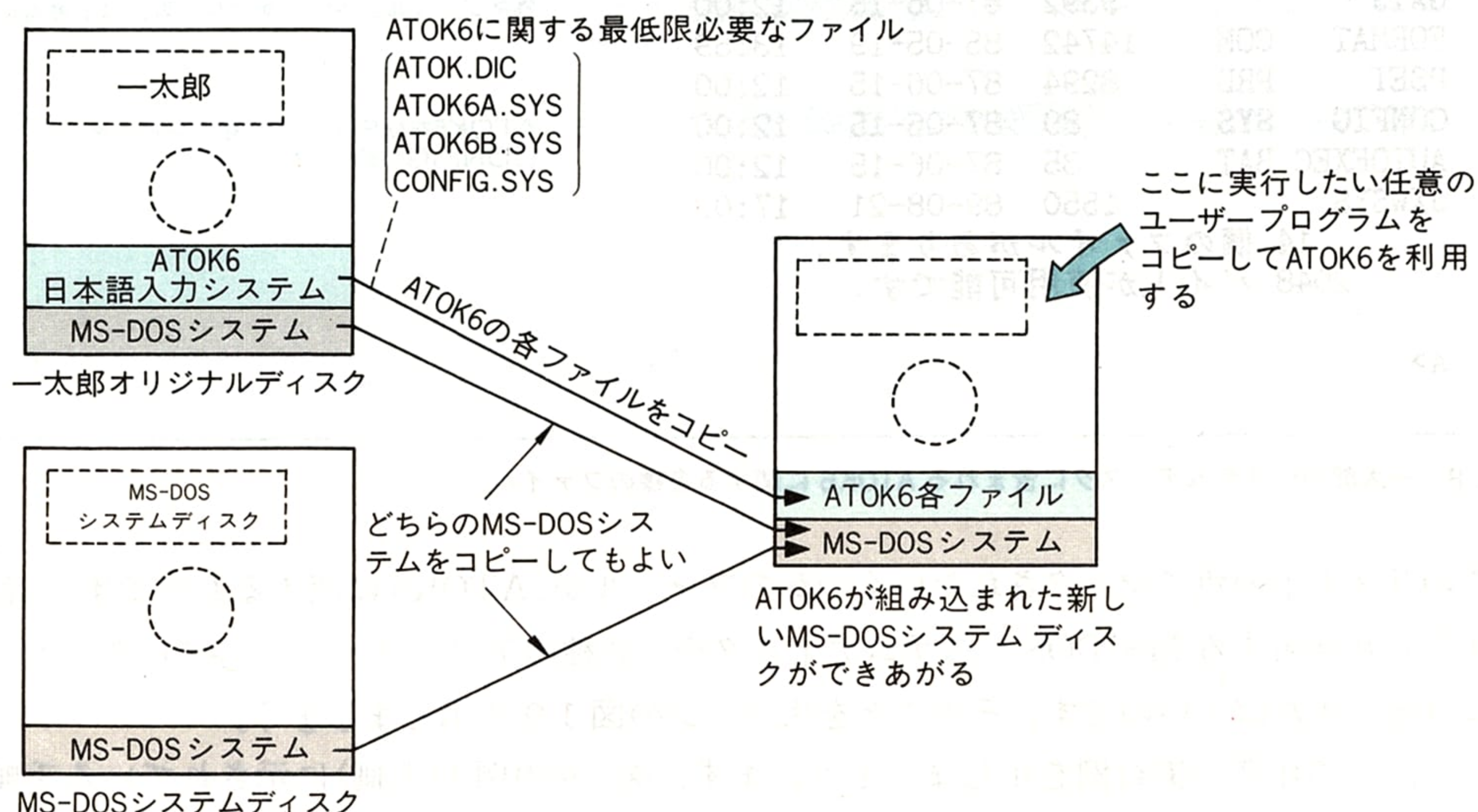
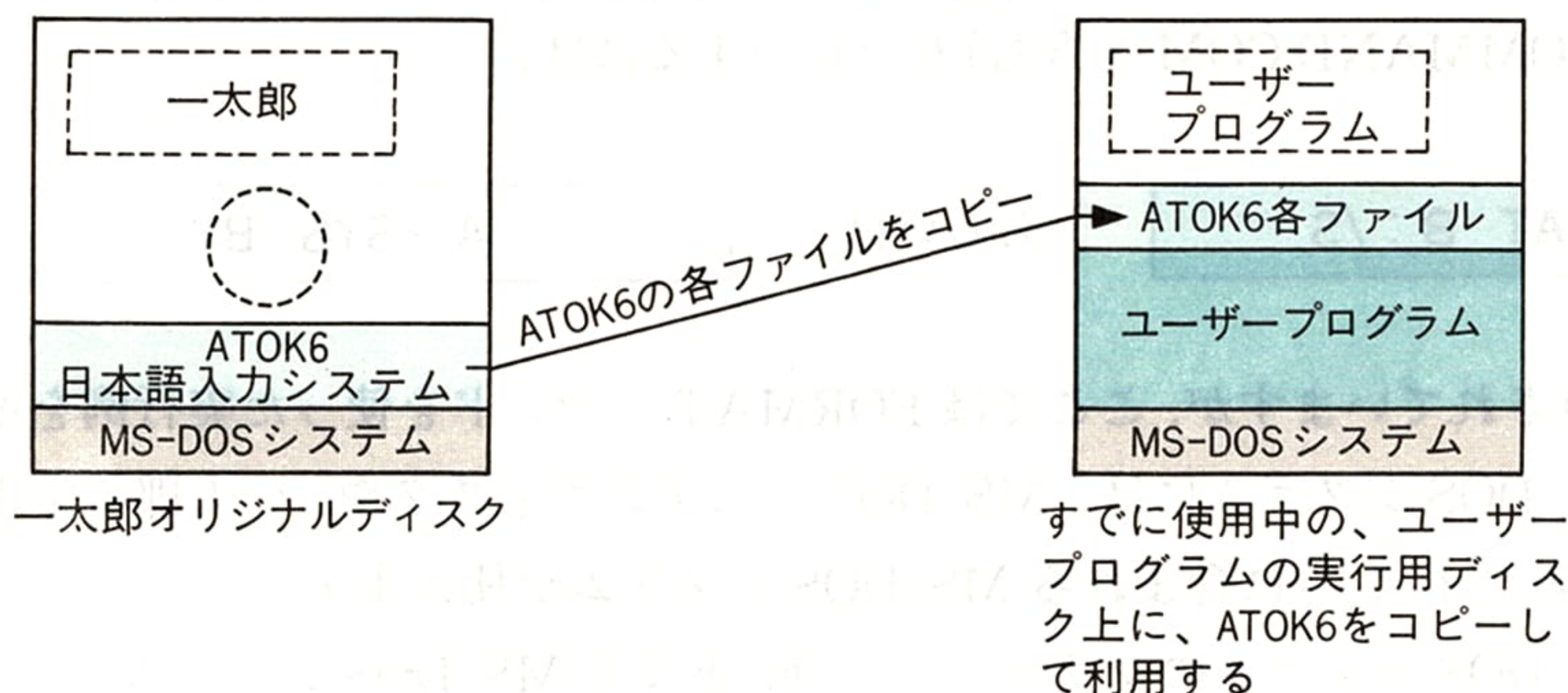



図1.9　ATOKを移植する2つの方法

テムをコピーします。そのあと、19 ページの図 1.8 に示した ATOK6 関係の各種のファイルを COPY コマンドでコピーすれば、ひとまず作業は終わりです。

ここまでの作業の実行例を次に示します。

```
A>FORMAT B:/S ⏎ …………ドライブB：のディスクをフォーマットし、ドライブA：のMS-DOSシステムをコピーする
Format Version 4.10

        ⋮（途中省略）

別のディスクをフォーマットしますか(Y/N) N ⏎ …………COPYコマンドを終了する
A>COPY ATOK.DIC B: ⏎
        1 個のファイルをコピーしました.

A>COPY ATOK6A.SYS B: ⏎
        1 個のファイルをコピーしました.

A>COPY ATOK6B.SYS B: ⏎                     ドライブA：の一太郎のシステムディスク上のこれらの
        1 個のファイルをコピーしました.    ファイルを、ドライブB：上にコピーする。

A>COPY GAIJ B: ⏎ ……外字を使用しなければ省略してよい
        1 個のファイルをコピーしました.

A>COPY CONFIG.SYS B: ⏎
        1 個のファイルをコピーしました.

A>DIR B: ⏎ …………結果の確認

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

COMMAND  COM    24931  88-07-13   0:00 …………「FORMAT /S」コマンドでコピーされたもの
                                          （MS-DOSシステムもコピーされている）
ATOK     DIC   454144  87-07-24  13:37
ATOK6A   SYS    54265  87-06-15  12:00
ATOK6B   SYS    21372  87-06-15  12:00   COPYコマンドでコピーされたもの
GAIJ             9392  87-06-15  12:00
CONFIG   SYS       89  87-06-15  12:00
        6 個のファイルがあります.
   587776 バイトが使用可能です.

A>
```

図 1.10　ATOK6 を組み込んだ MS-DOS システムディスクの作成

以上の作業で、ATOK6 に関するすべてのファイルがコピーされた、MS-DOS システムディスクがドライブ B：上にできあがりました。ATOK6 による日本語入力が行える MS-DOS システムディスクが用意できたわけです。

このシステムディスクをドライブA：にセットし、リセットボタンを押せばMS-DOSが立ち上がり、ATOK6による日本語入力が機能するはずですが、その前に、**CONFIG.SYSファイル**の内容を確認しておきましょう。この内容によっては、図1.10に示した各種のファイルがコピーされていても、ATOK6は動作しません。では、図1.10でコピーされたCONFIG.SYSファイルの内容(一太郎のオリジナルディスクに付属しているもの)をタイプアウトしてみましょう。

```
A>TYPE CONFIG.SYS
FILES=12
BUFFERS=5
DEVICE=CACHE.SYS  /C=4
DEVICE=ATOK6A.SYS /T=1
DEVICE=ATOK6B.SYS

A>
```

一太郎のオリジナルディスクに付属のCONFIG.SYSファイルの内容をタイプアウトする

この3行はなくてもATOK6は動作する(3.1章を参照)

この部分はATOK6の初期入力モードを設定するための環境パラメータであるが、当面は省略してよい。パラメータを省略した場合は、標準状態となる。くわしくは後述。また、ユーティリティプログラム「ATUT.EXE」(一太郎のユーティリティディスクに含まれている)を実行すれば、これらの任意のパラメータが自動的に設定される。

この2行は絶対に必要

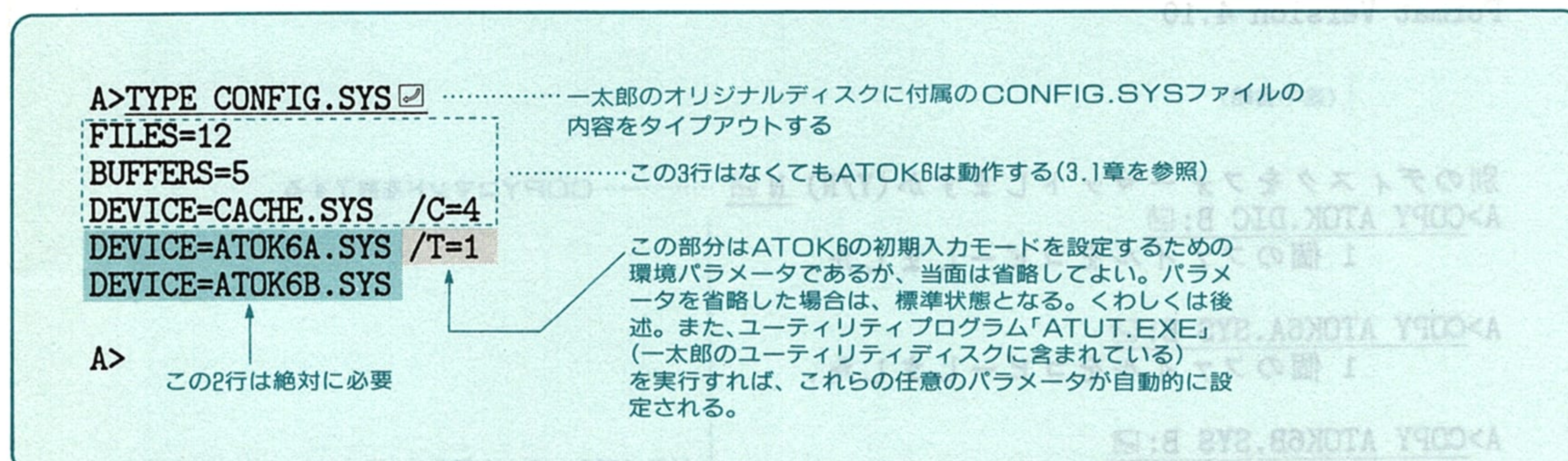



図1.11　ATOK6をMS-DOSシステムに組み込むために必要なCONFIG.SYSファイルの内容

ここで示されているCONFIG.SYSファイルの内容のうち、

DEVICE＝ATOK6A.SYS
DEVICE＝ATOK6B.SYS

の部分は、ATOK6が機能するためには絶対に必要です。ATOK6を組み込んだ各自のディスク上のCONFIG.SYSファイルをタイプアウトして確認してみてください。この部分が記述されていない場合、エディタを使ったり、後述する方法などでCONFIG.SYSファイルの内容を書き換えてください。

なお、CONFIG.SYSファイルの機能については、前節や、3.1章で触れていますが、そのファイルの内容で指示される、各種のデバイスドライバのMS-DOSシステムへの組み込みや、環境の設定が、MS-DOSの起動時に実行されるものと考えればよいでしょう。上に示した2行は、ATOK6のデバイスドライバ(2つに分れている)を、コンピュータのメモリ上の起動したMS-DOSシステムに組み込むことを指示するものです。

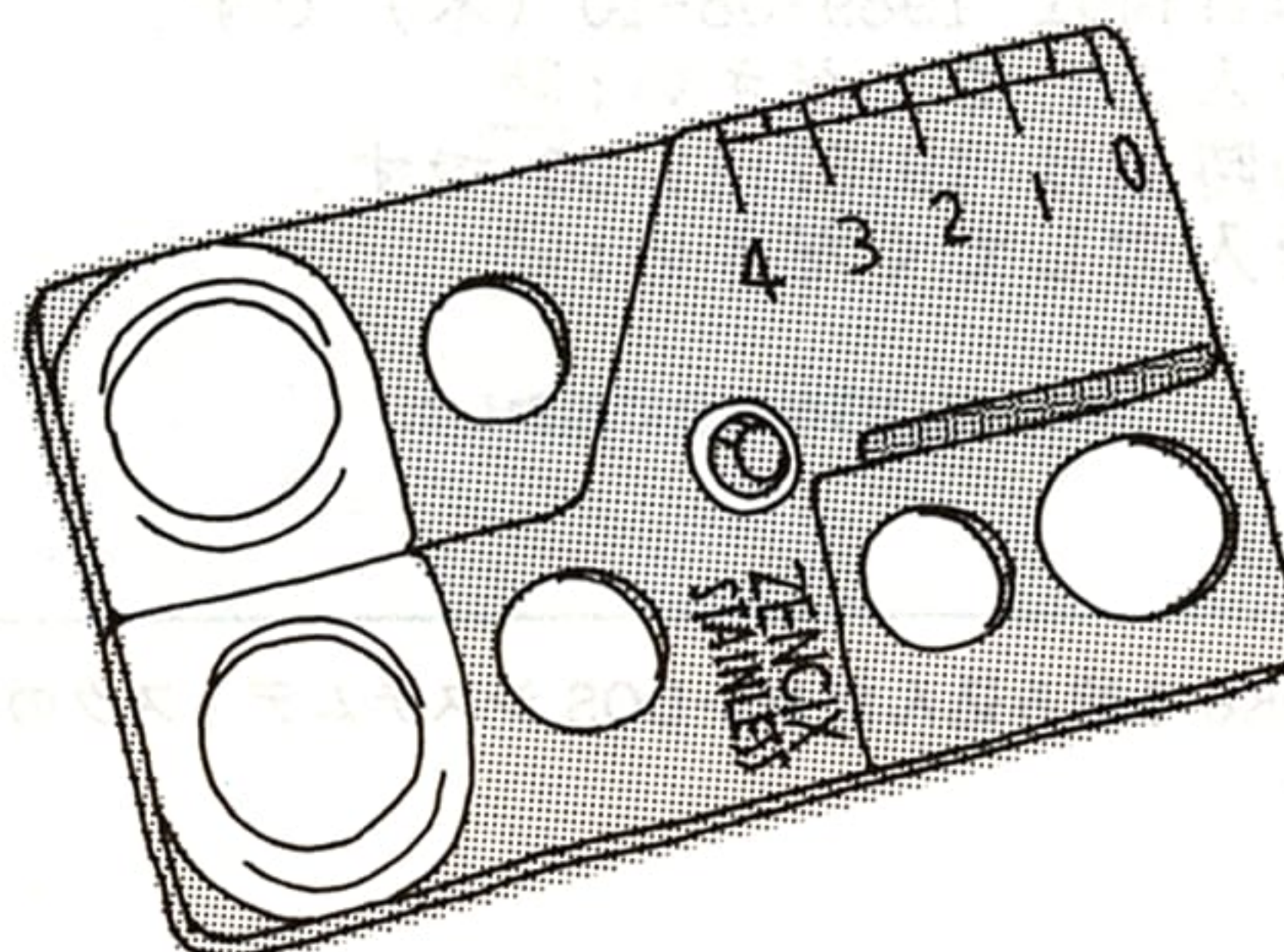
CONFIG.SYSファイルは、テキストファイル(文字ファイル)です。これを作成したり変更するには、MS-DOSのシステムディスクに標準装備のエディタプログラム「EDLIN.EXE(.COM)」(5章を参照)や、各種のエディタ(ワープロでもよい)を利用しますが、CONFIG.SYSファイルのような、短い簡単な内容であれば、内蔵コマンドのCOPYコマンドを使って、新たに作成してもよいでしょう。その方法は本書の5.1章でも解説していますが、ここでの実行例を次に示します。

```
A>COPY CON CONFIG.SYS⏎ ............COPYコマンドを使って、CONFIG.SYSファイルを作成する
FILES=10⏎     } これらはなくてもATOK6の動作は可能(3.1章を参照)
BUFFERS=20⏎   }
DEVICE=ATOK6A.SYS⏎ } この2行は絶対に必要
DEVICE=ATOK6B.SYS⏎ }
^Z⏎ ............CTRL+Zを入力した後、⏎する
        1 個のファイルをコピーしました. ............この時点で、キー入力された文字の内容の
                                                   ファイルが作成されMS-DOSにもどる

A>TYPE CONFIG.SYS⏎ ............作成されたファイル「CONFIG.SYS」をタイプアウトして確認する
FILES=10
BUFFERS=20
DEVICE=ATOK6A.SYS   このように作成されている
DEVICE=ATOK6B.SYS

A>
```

図 1.12　COPY コマンドを使って CONFIG.SYS ファイルを作成する

このように、COPY コマンドを使って、新しい CONFIG.SYS ファイルが簡単に作成されました。なお、この方法でファイルを作成する場合、作成しようとするファイルと同じファイル名のファイルがすでに存在していると、もとあったファイルは削除され、新しいファイルに置き換わってしまいますので注意してください。つまり、もとあった CONFIG.SYS ファイルは削除されてしまいます。

すでにある CONFIG.SYS ファイルを利用して、その一部を変更したり、新しい行を追加したりする場合は、エディタやワープロを使わなければなりません。それを COPY コマンドで行う場合は、その全体を書き直さなければならないわけです。

さて、CONFIG.SYS ファイルの内容を確認し、上の 2 行が含まれていれば ATOK6 による日本語入力が行えます。ここで作成したディスクには、MS-DOS の本体と ATOK6 が含まれているだけですので、各種の実務には、それぞれのアプリケーションプログラムや必要な各種のファイルを、このディスク上にコピーした上で使用することになります。

アプリケーションプログラムの常用ディスクにATOKを組み込む

次に、20ページの図1.9の例2(下側)の方法で、ATOK6の各ファイルを目的のディスクにコピーする実行例を示しましょう。この場合も考え方は同じです。具体的には、21ページの図1.10に示した実行例のCOPYコマンドを、目的のアプリケーションプログラムの常用ディスクに対して実行し、ATOK6の各ファイルをコピーすればよいのです。もしそのディスクに、すでにほかのFEPが存在している場合は、ディスク容量を考慮して、事前にそのFEPに関するファイルを削除しておきます(ハードディスクなどで、容量に十分な余裕があればその必要はない)。とくに辞書ファイルは大容量であるため、フロッピーディスクの場合は、2種類のFEPを同時に収容しておくことはできません。

ここでもCONFIG.SYSファイルの内容は重要です。すでに存在しているCONFIG.SYSファイルは、別のFEPに対応した内容であるかもしれません。その内容をタイプアウトして確認し、エディタやワープロを使って書き換えるか、図1.12に示したように作成し直さなければなりません。CONFIG.SYSファイルの内容には十分注意してください。

■ ATOK6による日本語入力の操作法

では、例1や例2の方法で作成したATOK組み込みのシステムディスクをドライブA：にセットし、リセットボタンを押して、MS-DOSを再起動してみましょう。ATOK6のシステムは、MS-DOSの起動時に、デバイスドライバとしてメモリ上のMS-DOSシステムに組み込まれます。**重要！**

```
(リセットボタンを押す)
                         (ATOK6は、約100Kバイトのメインメモリを消費します)
NEC PC-9800 Series Personal Computer

ﾏｲｸﾛｿﾌﾄ MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30
Copyright 1981,88 Microsoft Corp. / NEC Corporation

自由文変換システム　ＡＴＯＫ６ ver 1.2  } ATOK6日本語入力システムが、MS-DOSシステムに
Copyright 1986,87 (株)ジャストシステム  } 組み込まれたことを示すオープニングメッセージ

Command ﾊﾞｰｼﾞｮﾝ 3.30
現在の日付は 1989-08-10 (木) です.
日付を入力してください:⏎
現在の時刻は 19:37:11.00 です.
時刻を入力してください:⏎

A> ……………MS-DOSが完全に起動した
```

図1.13　ATOK6を組み込んだMS-DOSシステムディスクの起動

MS-DOS のオープニングメッセージと、ATOK6 が組み込まれたことを示すメッセージが表示され、MS-DOS が立ち上がりました。これで ATOK6 による日本語入力が可能になりました。もし、前項(図 1.9 の例 2 の場合)で作成したディスクのアプリケーションプログラムが、自動スタートで立ち上がってしまった場合は、そのソフトを「終了」して MS-DOS におり、プロンプト「A>」の状態にしてください。

さて、ATOK6 が動作する環境が整ったとして、次は実際に日本語入力を行うための操作の基本を実習しましょう。

まず、FEP が機能する状態と、機能しない状態との切り替え操作を行います。つまり、日本語入力が可能な状態(**日本語入力モード**)と、日本語入力機能が働かない状態(**通常入力モード**)との切り替えであり、FEP の ON／OFF と考えればよいでしょう。

日本語入力モードにはいるには、CTRL ＋ XFER を入力します(CTRL キーを押しながら XFER キーを押す)。日本語入力モードから、通常入力モードにもどるには、再度 CTRL ＋ XFER を入力します。

日本語入力モード⇄通常入力モードの切り替えは CTRL ＋ XFER を入力

日本語入力モードにはいった時点の画面を次に示します。

A>▮ ←スクリーンカーソル
入力ラインカーソル
連文節変換モード
ローマ字→漢字変換モード
連ローマ字漢字
日本語入力エリア(全角30文字入力可能)
現在の入力モードの表示

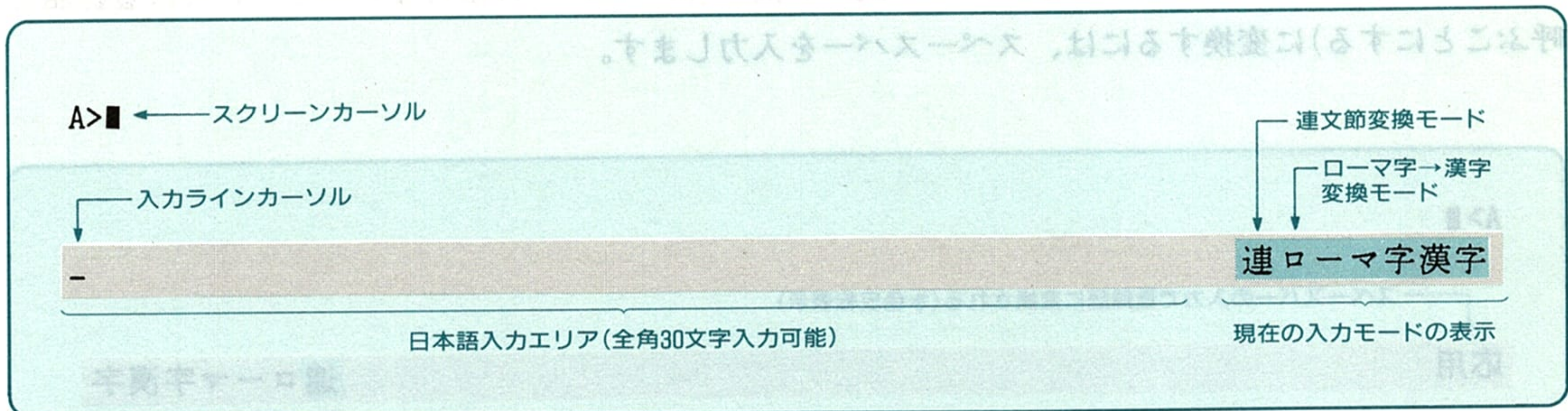



図 1.14　ATOK6 の日本語入力モードにはいった初期画面

画面最下段の日本語入力ラインの右端に、**ローマ字漢字**と表示されているように、ATOK6 では、日本語入力モードにはいった時点の標準設定は、ローマ字による入力モードになっています(CONFIG.SYS ファイルの記述により任意の設定が可能。後述)。この入力方式については、プロのタイピストを志望する人を除き、FEP の種類に関係なくローマ字による入力を強くお勧めします。

ではこの状態で、とりあえず「応用 MS-DOS」という文字列を入力してみましょう。 カナ キーがOFF であることを確認した後、「OUYOU」とキー入力します(カナ キーが OFF でなければローマ字(英字)入力はできない)。英字を入力する際の、大文字／小文字を選択する CAPS は、ON でも OFF でもどちらでもかまいません。英字そのものを入力する際に、大文字／小文字のどちらの頻度が多いかによって、使いやすい方を選択しておけばよいでしょう。ミスタイプした場合は、 BS でカーソルの左の1文字を、 DEL でカーソル上の1文字を、また ESC で入力文字全体を削除することができます。

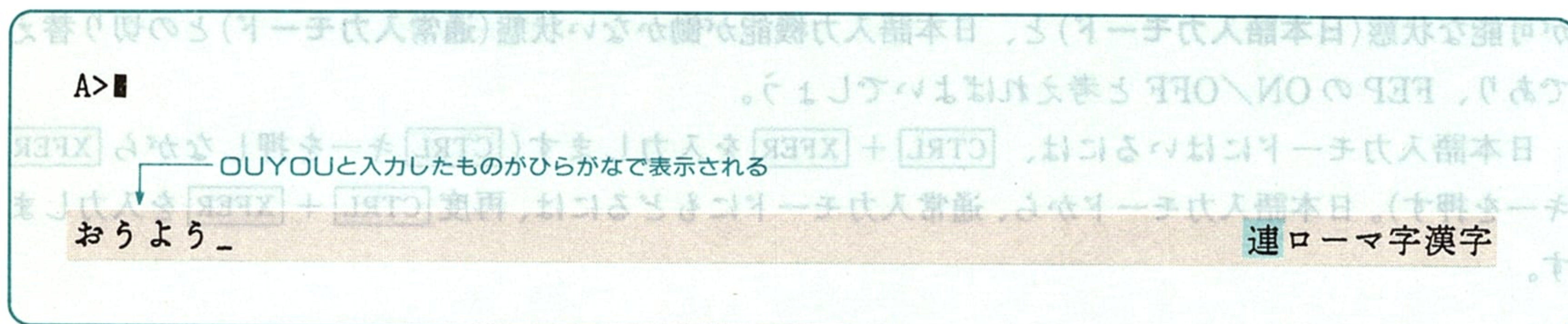



図 1.15　ローマ字による読みの入力

このようにローマ字で入力したものが、その場で自動的にひらがなに変換されながら、日本語入力ラインに「おうよう」と入力されます。この読みを漢字(漢字だけとは限らないので、以後は**登録語**と呼ぶことにする)に変換するには、スペースバーを入力します。

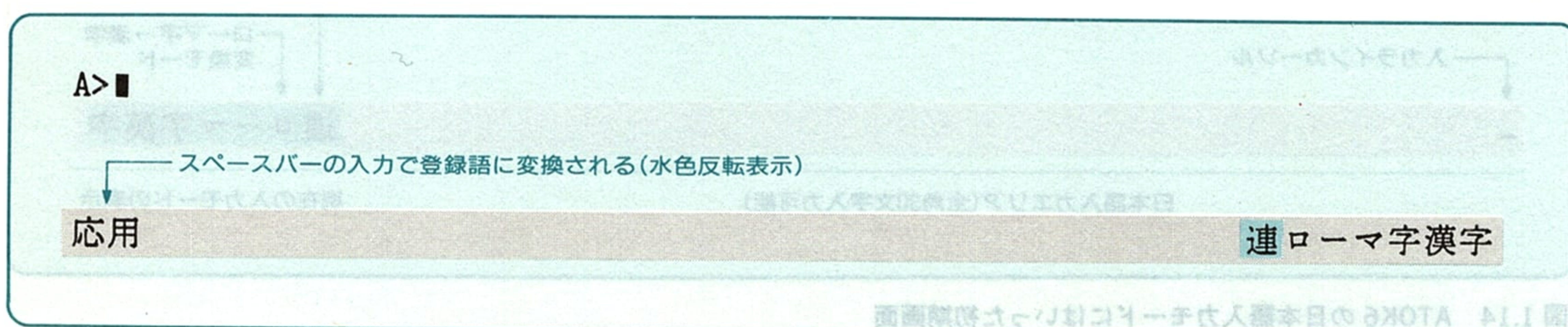



図 1.16　スペースバーの入力により漢字(登録語)に変換される

このように辞書ファイルに登録されている語のなかから、「応用」が取り出されています。この場合はこれで正解ですが、目的とする登録語に変換されなかった場合は、さらにスペースバーを入力することにより、同じ読みの別の登録語を取り出すことができます(もちろん登録されていればの話ですが)。このようにスペースバーを次々と押していけば、同じ読みの登録語が次々と出てきますが、いくつか前に出てきた登録語にもどしたい場合には、カーソル移動キーの ↑ をその回数だけ入力します。

さて、日本語入力ライン上で、目的の文字列への変換ができたなら、続けて次の語の読みを入力するか、あるいは[↓]または[⏎]を入力することにより、それまでに入力した文字列が、日本語入力ライン上からスクリーンカーソルの位置に移動します。ここでの実行例におけるスクリーンカーソルは、MS-DOS のプロンプト「A＞」の位置ですが、実際には各種のビジネスソフトや、エディタなどの入力画面上であるわけです（日本語入力ラインを使わず、直接スクリーンカーソルの位置に入力することも可能。本節の最後で述べる CONFIG.SYS ファイルに記述する ATOK デバイスドライバのパラメータ「/E」を参照）。

「応用」と入力できたら、次は「MS-DOS」とキー入力します。この場合の綴りは、たまたまそのアルファベットの並びから、「ローマ字ーかな変換」されずに英字綴りのままですので、そのまま[⏎]を入力することにより、入力文字がスクリーンカーソルの位置に移ります。ATOK のローマ字入力は、3 文字以内に何らかの文字（ひらがな）に変換できなければ、その後の「ローマ字ーかな変換」は行われません。ところが、「DOS」とキー入力した場合などは、「どS」というぐあいに変換されます。このような場合は、[f･9]を入力することにより、キー入力した時点の英字綴りの文字「DOS」にもどすことができます。

A>応用▮

移動

次の語「MS-DOS」のMを入力したと同時に、「応用」の語がスクリーンカーソルの位置に移動した。[↓]または[⏎]の入力によっても移動する

ＭＳ－ＤＯＳ_　　連ローマ字漢字

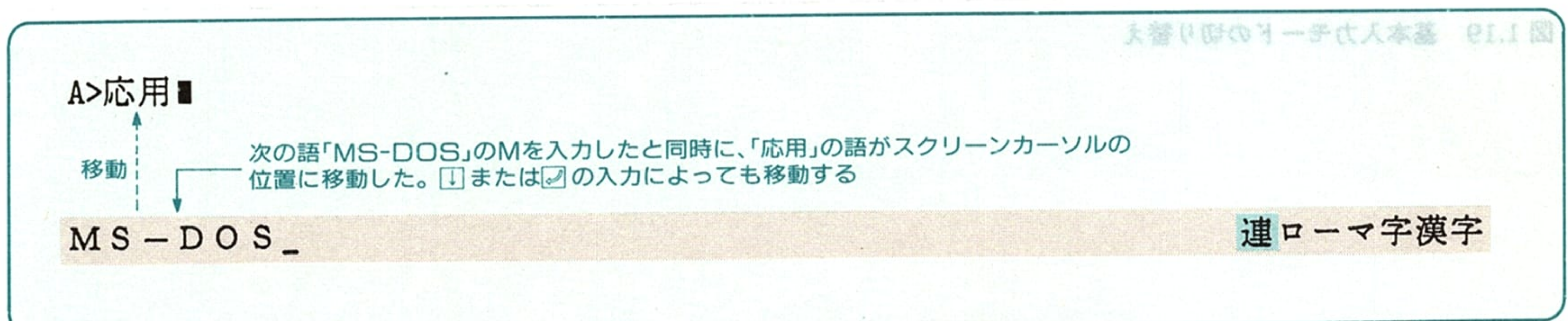



図 1.17　日本語入力ラインに入力された目的の語を、スクリーンカーソルの位置に移す

以上の操作によって、目的の語「応用 MS-DOS」を入力することができました。その画面を次に示します。

A>応用ＭＳ－ＤＯＳ▮

[⏎]または[↓]の入力により移動

_　　連ローマ字漢字

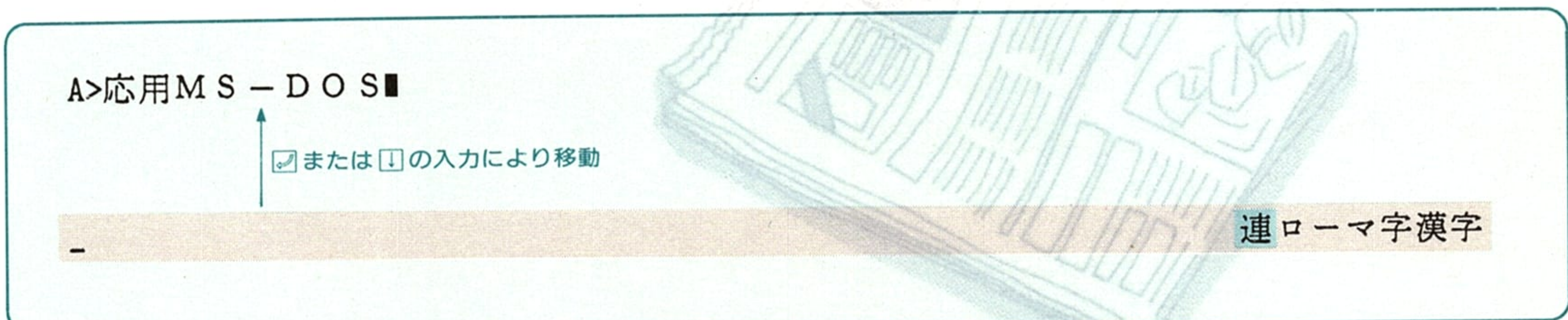



図 1.18　リターンキーの入力によって入力ライン上の文字列が、スクリーンカーソルの位置に移される

入力モードの選択

ATOK6には、5つの基本的な入力モードがあり（ATOK7では4つ）、それらを［f·10］により自由に選択することができます。選択されているモードは、日本語入力ラインの右端の表示によって知ることができます。その入力モードの種類を次に示します。［f·10］を入力することにより、入力モードが図に示す順序でロータリー式に切り替わります。

（基本入力モードの切り替え）

f·10

連ローマ字漢字 ①

連カナ漢字 ②

半角英数カナ ③

コード JIS ④

JIS 【 2121 】 、。，．・：；？！　記号 ⑤

連ローマ字漢字 ①

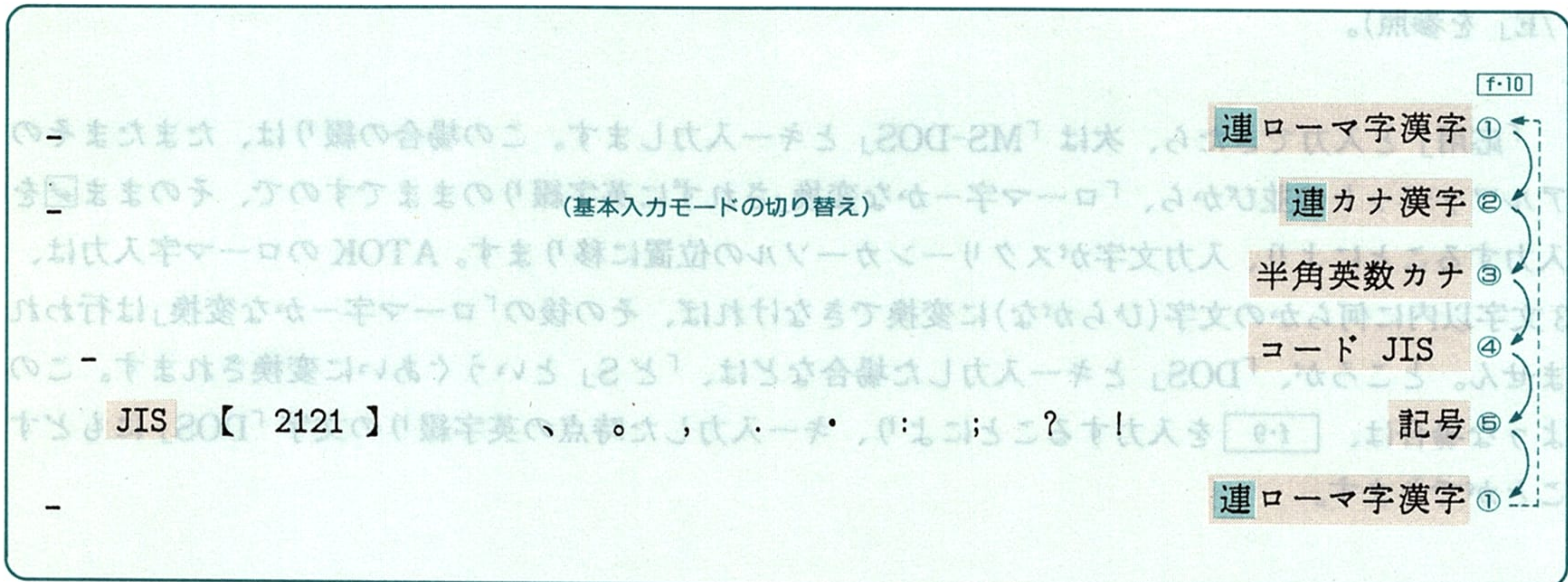
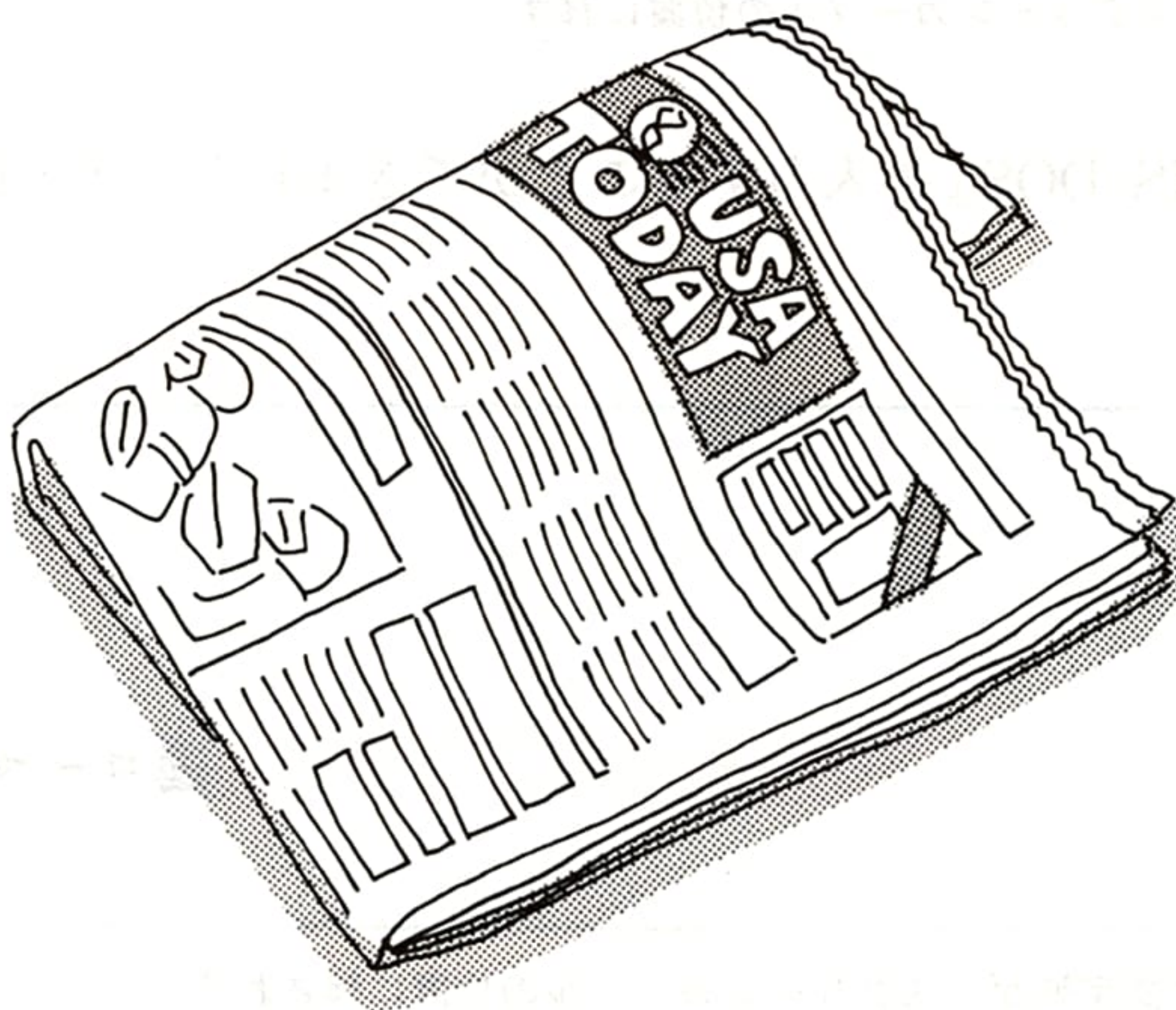



図 1.19　基本入力モードの切り替え

これらの基本入力モードのうち、④の「コード」、および⑤の「記号」の各モードは、漢字コードによって目的の文字(単字)を検索するもので、漢字コードは、**シフトJISコード**、**JISコード**、**区点コード**の3種類から選択できます(漢字コードについては2章を参照)。漢字コードの選択は、④または⑤のモードにおいて、SHIFT＋f·4（ATOK7ではSHIFT＋f·7）を入力することにより、次の図に示すように3種類のコードがロータリー式に切り替わります。

④の「コード」モードにおける漢字コードの選択と、⑤の「記号」モードとは連動しており、両者は常に同一モードになります。この④、⑤モードにおける各種漢字コードを使った単字変換の実行例を次に示します。

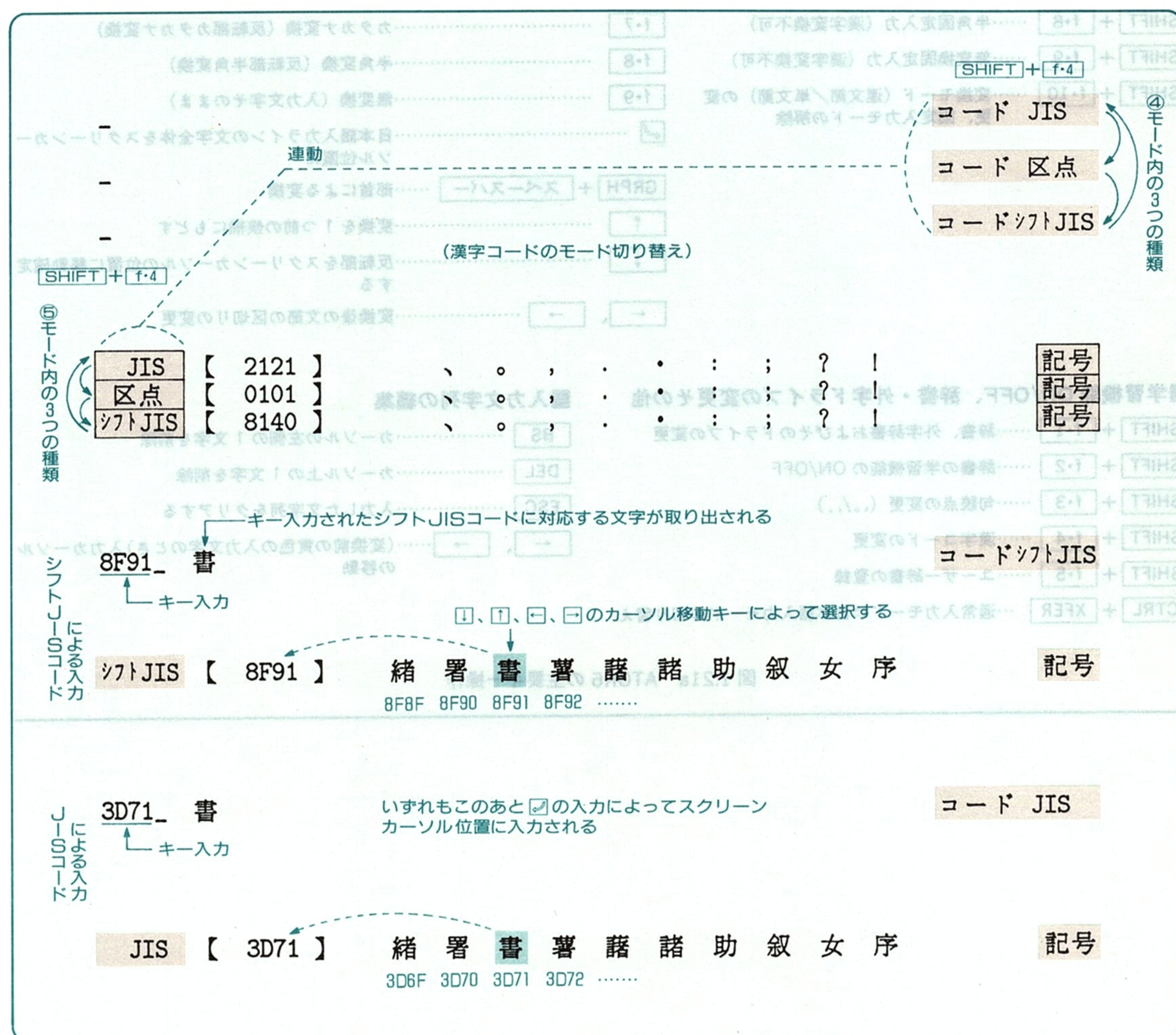
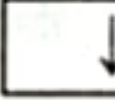
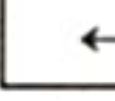
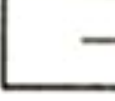



図1.20　各種漢字コードの切り替えと単字変換

各種変換機能の基本操作

ここで、ATOK6の基本的なキー操作と、注意しなければならないローマ字入力の綴り方などを表にまとめて示します。

■入力・変換モードの切り替え

- [f･10] ……入力モード（ローマ字漢字→カナ漢字→半角英数→コード入力→記号入力）の切り替え
- [SHIFT]＋[f･6] ……ひらがな固定入力（漢字変換不可）
- [SHIFT]＋[f･7] ……カタカナ固定入力（漢字変換不可）
- [SHIFT]＋[f･8] ……半角固定入力（漢字変換不可）
- [SHIFT]＋[f･9] ……無変換固定入力（漢字変換不可）
- [SHIFT]＋[f･10] ……変換モード（連文節／単文節）の変更、固定入力モードの解除

■変換

- [スペースバー] ……漢字・熟語等の登録語への変換。次候補の表示および選択
- [XFER] ……漢字・熟語等の登録語への変換。次候補の表示（選択は番号またはスペースバーで）
- [f･6] ……ひらがな変換（反転部ひらがな変換）
- [f･7] ……カタカナ変換（反転部カタカナ変換）
- [f･8] ……半角変換（反転部半角変換）
- [f･9] ……無変換（入力文字そのまま）
- [↵] ……日本語入力ラインの文字全体をスクリーンカーソル位置に
- [GRPH]＋[スペースバー] ……部首による変換
- [↑] ……変換を１つ前の候補にもどす
- [↓] ……反転部をスクリーンカーソルの位置に移動確定する
- [←]、[→] ……変換後の文節の区切りの変更

■学習機能 ON/OFF、辞書・外字ドライブの変更その他

- [SHIFT]＋[f･1] ……辞書、外字辞書およびそのドライブの変更
- [SHIFT]＋[f･2] ……辞書の学習機能のON/OFF
- [SHIFT]＋[f･3] ……句読点の変更（、。/，．）
- [SHIFT]＋[f･4] ……漢字コードの変更
- [SHIFT]＋[f･5] ……ユーザー辞書の登録
- [CTRL]＋[XFER] …通常入力モード⇄日本語入力モードの切り替え

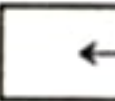
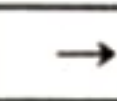
■入力文字列の編集

- [BS] ……カーソルの左側の１文字を削除
- [DEL] ……カーソル上の１文字を削除
- [ESC] ……入力した文字列をクリアする
- [←]、[→] ……（変換前の黄色の入力文字のとき）入力カーソルの移動

図 1.21a　ATOK6の主要キー操作

通常入力モード

CTRL + XFER

日本語入力モード

①→ ローマ字漢字モード →(f・10)→ カナ漢字モード →(f・10)→ 半角英数カナモード →(f・10)→ コード入力モード →(f・10)→ 記号入力モード →(f・10)→ ①

ローマ字漢字モード

キー入力
カナキーOFF

f・6 → ひらがな変換
f・7 → カタカナ変換
f・8 → 半角英数 カナ変換
f・9 → 無変換 英数字
スペースバー XFER → 登録語変換

カナ漢字モード

キー入力
カナキーON

f・6 → 同　左
f・7 → 同　左
f・8 → 半角カナ変換
f・9 → 無変換 カタカナ
スペースバー XFER → 同　左

半角英数カナモード

キー入力
カナキーON/OFF

変換なし。
スクリーンカーソルの位置へ直接入力される

コード入力モード

JISモード
SHIFT + f・4
シフトJIS モード
SHIFT + f・4
区点モード
SHIFT + f・4

各コードの数値の入力

記号入力モード

カーソル移動キーによる選択入力

シフトJIS／JIS／区点の各モードは左に連動。モードの選択は左と同じ

図 1.21b　ATOK6 の変換モード

長　音	
ローマ字	RO－MAZI*
ユーザー	YU－ZA－
つまる音	
がっこう(学校)	GAKKOU
いっち(一致)	ITTI
あっ	ALTU
うっ	ULTU

＊－は「マイナス」または「ハイフン」記号

を、ん	
を	WO
ん	NN
しんい(真意)	SINNI
ほんしゃ(本社)	HONSYA
かな小文字	
ぁ	LA
ぃ	LI
ゅ	LYU

あ	あ A	い I	う U	え E	お O	ぁ LA	ぃ LI	ぅ LU	ぇ LE	ぉ LO
か	か KA	き KI	く KU	け KE	こ KO	きゃ KYA	きぃ KYI	きゅ KYU	きぇ KYE	きょ KYO
さ	さ SA	し SI SHI	す SU	せ SE	そ SO	しゃ SYA SHA	しぃ SYI	しゅ SYU SHU	しぇ SYE SHE	しょ SYO SHO
た	た TA	ち TI CHI	つ TU TSU っ LTU	て TE	と TO	ちゃ TYA CYA CHA てゃ THA	ちぃ TYI CYI てぃ THI	ちゅ TYU CYU CHU てゅ THU	ちぇ TYE CYE CHE てぇ THE	ちょ TYO CYO CHO てょ THO
な	な NA	に NI	ぬ NU	ね NE	の NO	にゃ NYA	にぃ NYI	にゅ NYU	にぇ NYE	にょ NYO
は	は HA	ひ HI	ふ HU FU	へ HE	ほ HO	ひゃ HYA ふぁ FA ふゃ FYA	ひぃ HYI ふぃ FI FYI	ひゅ HYU ふ FU ふゅ FYU	ひぇ HYE ふぇ FE FYE	ひょ HYO ふぉ FO ふょ FYO
ま	ま MA	み MI	む MU	め ME	も MO	みゃ MYA	みぃ MYI	みゅ MYU	みぇ MYE	みょ MYO
や	や YA	い YI	ゆ YU	え YE	よ YO	ゃ LYA	ぃ LYI	ゅ LYU	ぇ LYE	ょ LYO
ら	ら RA	り RI	る RU	れ RE	ろ RO	りゃ RYA	りぃ RYI	りゅ RYU	りぇ RYE	りょ RYO
わ	わ WA	うぃ WI	う WU	うぇ WE	を WO					
ん	ん NN									
が	が GA	ぎ GI	ぐ GU	げ GE	ご GO	ぎゃ GYA	ぎぃ GYI	ぎゅ GYU	ぎぇ GYE	ぎょ GYO
ざ	ざ ZA	じ ZI JI	ず ZU	ぜ ZE	ぞ ZO	じゃ ZYA JA JYA	じぃ ZYI JYI	じゅ ZYU JU JYU	じぇ ZYE JE JYE	じょ ZYO JO JYO
だ	だ DA	ぢ DI	づ DU	で DE	ど DO	ぢゃ DYA でゃ DHA	ぢぃ DYI でぃ DHI	ぢゅ DYU でゅ DHU	ぢぇ DYE でぇ DHE	ぢょ DYO でょ DHO
ば	ば BA	び BI	ぶ BU	べ BE	ぼ BO	びゃ BYA	びぃ BYI	びゅ BYU	びぇ BYE	びょ BYO
ぱ	ぱ PA	ぴ PI	ぷ PU	ぺ PE	ぽ PO	ぴゃ PYA	ぴぃ PYI	ぴゅ PYU	ぴぇ PYE	ぴょ PYO
ヴ	ヴァ VA	ヴィ VI	ヴ VU	ヴェ VE	ヴォ VO					

表 1.1　ATOK6 におけるローマ字入力の綴り方

さきほど、「応用 MS-DOS」という文字列を入力してみましたが、その時の「OUYOU」というローマ字入力をもとに、ひらがな、カタカナ、半角カタカナ、登録語変換、無変換などの、各種の変換と、その基本操作について解説しましょう。この操作は、ローマ字入力モードの場合のみに限らず、カタカナキーによるカナ入力モードの場合においても、ほとんど共通ですので、ローマ字入力を使わない方も目を通しておいてください。

A>█ ←スクリーンカーソル

入力ラインカーソル

いずれの状態からも ESC の入力によりすべてがクリアされ、この初期状態にもどる

_　　連ローマ字漢字

いずれの状態からも BS の入力により、この最初の入力状態にもどる

おうよう_ ……OUYOUと読みを入力(黄色文字)　　連ローマ字漢字

どのような順序でもよい

オウヨウ ……f·7 を入力(水色反転)　　連ローマ字漢字

ｵｳﾖｳ ……f·8 を入力(水色反転)　　連ローマ字漢字

ＯＵＹＯＵ ……f·9 を入力(水色反転)　　連ローマ字漢字

ｏｕｙｏｕ ……同上(読みを英小文字で入力した場合)　　連ローマ字漢字

OUYOU ……f·8 を入力(水色反転)　　連ローマ字漢字

ouyou ……同上(読みを英小文字で入力した場合)　　連ローマ字漢字

応用 ……スペースバーを入力(水色反転)　　連ローマ字漢字

1.応用　2.逐　3.負　4.逐 ……再度スペースバーを入力(スペースバーまたは番号で目的の語を選択する)　残　2　　連ローマ字漢字

おうよう_ ……BS を入力。最初の状態にもどる(黄色文字)　　連ローマ字漢字

応用 ……スペースバーを入力(水色反転)　　連ローマ字漢字

A>応用

上の状態から ⏎ の入力により、スクリーンカーソルの位置に移る

_　　連ローマ字漢字

A>応用

上の状態から次の文字の入力により、スクリーンカーソルの位置に移る

M_　　連ローマ字漢字

図 1.22　各種変換機能の基本操作

入力した「読み」の、カタカナ、ひらがな、半角、無変換などへの変換作業——つまり、辞書ファイルを検索しない(登録語に変換しない)変換作業は、ファンクションキーの［f･6］〜［f･9］で行います。一方、漢字変換など、辞書ファイルを検索して登録語を取り出すものは、スペースバー、あるいは［XFER］で行います。

前ページの図 1.22 には、入力した「読み」が、それぞれのファンクションキーや、スペースバーによってさまざまに変換される様子が示されています。変換により、入力ラインカーソル「_」が消え、水色反転になることにも注目してください。水色反転区間は、再度スペースバーにより、次候補への変換や再変換が可能であることを示していますが、くわしくは後述します。

ワープロの場合もそうですが、日本語入力の操作をマスターするには、とにかく実際に使ってみることです。どんどんキー入力して、いろいろなことを試みてください。

［BS］、［ESC］による入力文字の削除

日本語入力ラインに入力された「読み」は、［BS］によりカーソルの左側の文字を１文字ずつ、また［DEL］によりカーソル上の文字を１文字ずつ、あるいは［ESC］により入力文字全体を削除することができます。そのときの日本語入力ラインの変化を次の実行例で示しましょう。

入力ライン	説明	状態
おうよう_	OUYOUと読みを入力(黄色文字。カーソルあり)	連ローマ字漢字
オウヨウ	［f･7］を入力(水色反転。カーソルなし)	連ローマ字漢字
おうよう_	［BS］を入力(黄色文字。カーソルあり)	連ローマ字漢字
応用	スペースバーを入力(水色反転。カーソルなし)	連ローマ字漢字
おうよう_	［BS］を入力(黄色文字。カーソルあり)	連ローマ字漢字
おうよ_	［BS］を入力(黄色文字。カーソルあり)	連ローマ字漢字
おう_	［BS］を入力(黄色文字。カーソルあり)	連ローマ字漢字
_	［ESC］を入力。すべての文字がクリアされ初期状態にもどる	連ローマ字漢字

いずれの状態からも、［ESC］の入力により、すべてがクリアされ初期状態にもどる

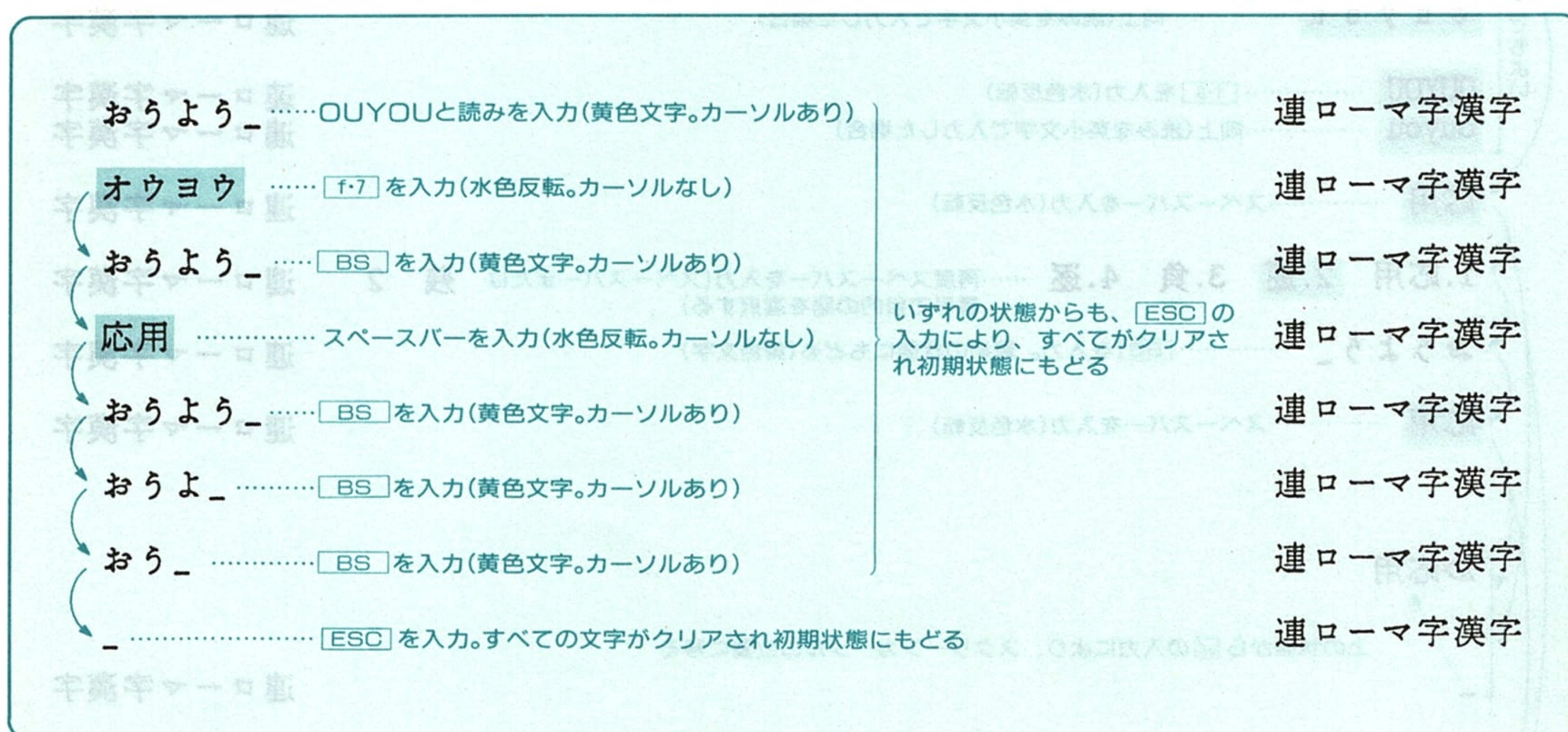



図 1.23　［BS］および［ESC］による入力文字の削除

カタカナキーによる入力

カナ キーをONにして、ひらがなの「読み」を直接カタカナキーで入力する場合は、図1.19の②の「カナ漢字」モードで行います(ATOK7では、SHIFT ＋ f･10 の入力モードの変更で行います)。各種の変換操作は、前述の「ローマ字変換」モードの場合と同じです。ただし「カナ漢字」モードの場合は、当然ながら無変換の f･9 キーを押しても英字にはなりませんので、英字の入力は、「カナ」をOFFにして行います。

「半角英数カナ」モードでの入力

半角文字(英数字、カナ、キーボード上の記号など)を入力するには、ATOK6を使わずに通常の入力モードで行えばよいのですが、ATOK6からは、図1.22の例にあるように、「ローマ字漢字」モードにおいて全角文字で入力したものを半角に変換する方法と、これから述べる「半角英数カナ」モードで直接入力する方法とがあります。

「半角英数カナ」モードは、図1.19の③のモードであり、半角の英数字(カナ キーをOFFにして入力)およびカタカナ(ONにして入力)を、スクリーンカーソルの位置に直接入力するものであり、他への変換はいっさいできません(ただしATOK7では、「連R漢」モードで入力した半角の英字を、XFER で、かなー漢字変換できる)。このモードで入力された文字は、ATOK6を使わない通常入力モードでキー入力したものと同等に、MS-DOSや、アプリケーションプログラムの各種のコマンドのキー入力として使うことができます。

では一例として、「半角英数カナ」モードで(ATOK7の場合は、「半角」モード)、MS-DOSのDIRコマンドを実行した場合を示しましょう。

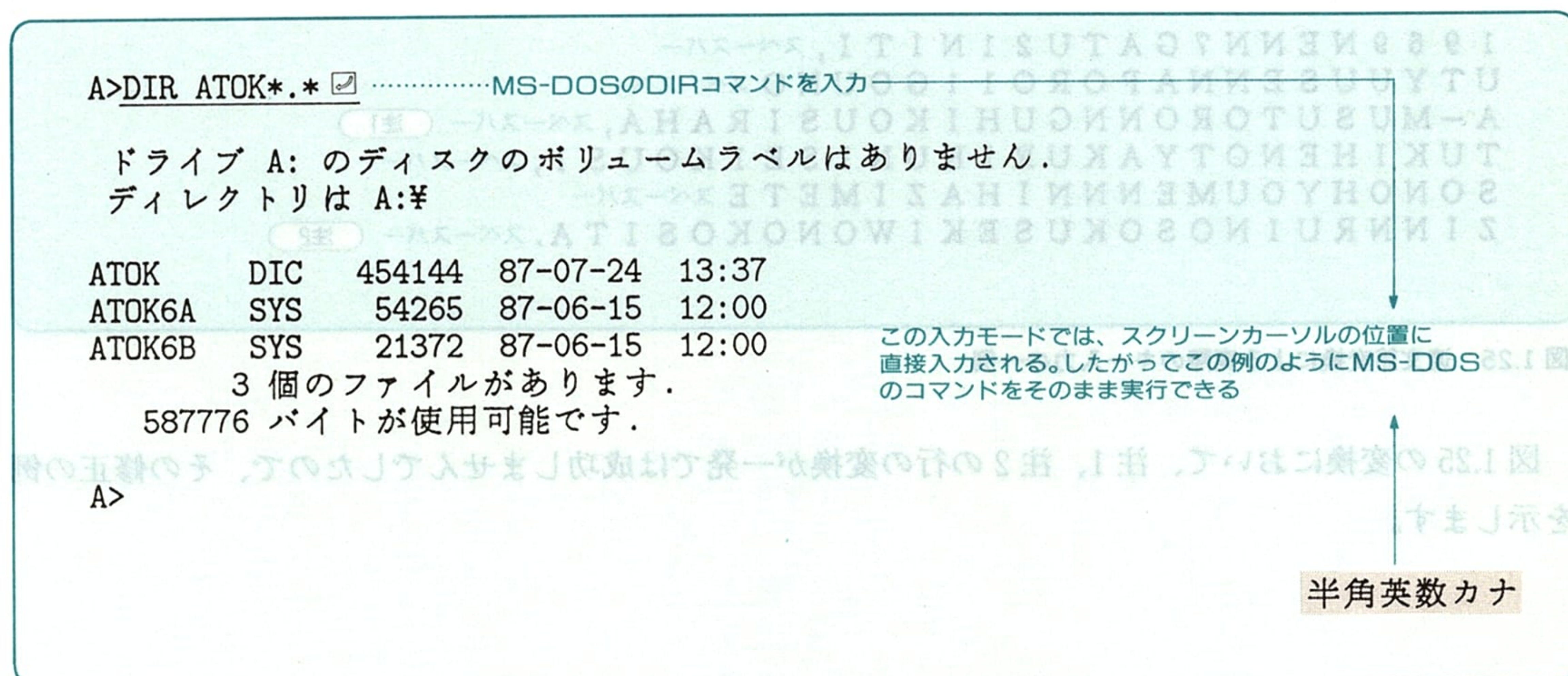
```
A>DIR ATOK*.*

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

ATOK      DIC   454144  87-07-24  13:37
ATOK6A    SYS    54265  87-06-15  12:00
ATOK6B    SYS    21372  87-06-15  12:00
        3 個のファイルがあります.
   587776 バイトが使用可能です.

A>
```


図1.24　「半角英数カナ」モードでのMS-DOSコマンドの実行

■ 連文節変換の入力例

ATOK6の連文節変換により、ワープロのような感覚で文を入力する一例を示しましょう。次の文を、スクリーンカーソルの位置(ここではMS-DOSのプロンプト、「A>」の後)に入力します。

1969年7月21日，宇宙船アポロ11号のアームストロング飛行士らは，
月への着陸に成功し，その表面に初めて人類の足跡を残した．

ここでの入力は、「ローマ字漢字」モードで行っています。文章を適切な連文節単位に区切りながら変換していくわけですが、この区切り方は、変換効率の面から重要なポイントになります。このノウハウは、それぞれの日本語入力システムによって異なり、これを簡単に解説することは困難です。使い込んだ経験から身につけるしかないでしょう。ただし、現在のすべての日本語入力システム(ワープロを含めて)について明確に言えることは、あまりいくつもの長い文節を一度に変換しようとせず、せいぜい2〜3の確実に変換できる範囲に区切って、こまめに変換していく方が結局は能率的だと言うことです。多くの文節からなる連文節の一部が正しく変換されなかった場合の修正は、非常にめんどうで時間がかかります。

さて、次に示すのは、この文章をローマ字で入力する場合のキー入力の一例を、変換単位に区切って示したものです。ここでは修正作業を実習するために、故意に長い文節に区切っていますが、実際には、なるべくこまめに区切ることをお勧めします。

```
1969NENN7GATU21NITI, スペースバー
UTYUUSENNAPORO11GOUNO スペースバー
A−MUSUTORONNGUHIKOUSIRAHA, スペースバー (注1)
TUKIHENOTYAKURIKUNISEIKOUSI, スペースバー
SONOHYOUMENNNIHAZIMETE スペースバー
ZINNRUINOSOKUSEKIWONOKOSITA. スペースバー (注2)
```

図1.25　連文節変換による実際のキー入力の一例

図1.25の変換において、注1、注2の行の変換が一発では成功しませんでしたので、その修正の例を示します。

注1

あーむすとろんぐひこうしらは，
水色反転
あーむすとロング飛行士らは，
水色反転
アームストロング飛行士らは，

スペースバー
f·7
または次の語の読みの入力
スクリーンカーソルの位置に移動

注2

じんるいのそくせきをのこした．
水色反転
人類の即席を残した．
水色反転
即席を残した．
水色反転
足跡を残した．

スペースバー
↓
スペースバー
または次の語の読みの入力
スクリーンカーソルの位置に移動

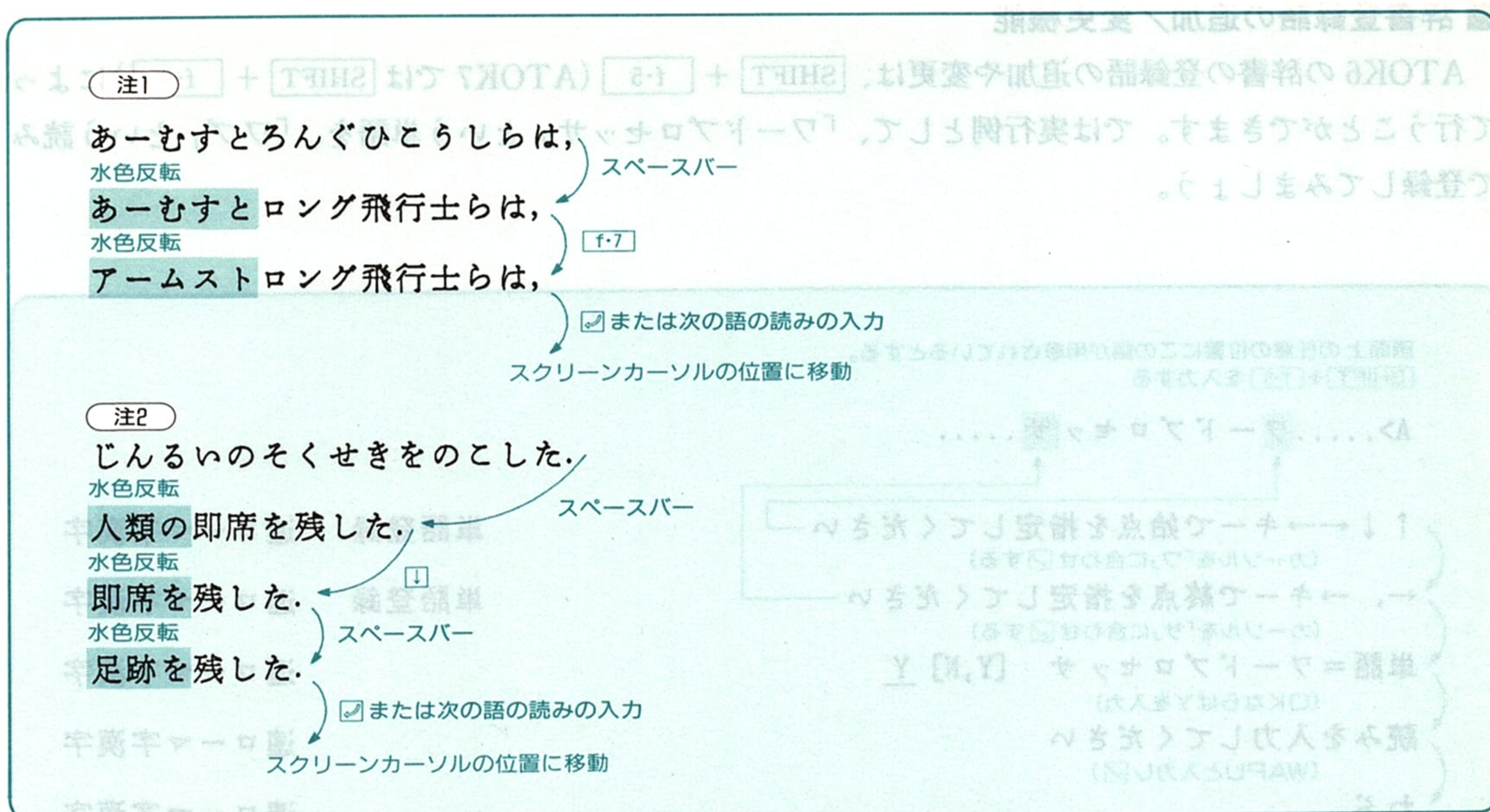



図 1.26a　連文節変換において、目的の語に変換されなかった場合の処理

このように、水色反転の部分は、スペースバーを再度入力することにより、次候補または再変換が可能です。表示された次候補群の選択は、スペースバーまたは [↑] で行います。また、文節の区切りを変更するには、[←]、[→] を入力することにより、1文字単位に任意に変更でき、黄色反転の区間が新たに1つの文節として変換の対象になります。目的の語への変更が成功したら、[↓] の入力により、水色や黄色の反転部の語をスクリーンカーソルに移動し、確定することができます。

A>飛行士らは，

ひこうしらは，　…………最初の入力
スペースバー
水色反転
非行白羽，　…………ひこうしらはのように文節の区切りを誤って変換された
←→←→←
黄色反転
ひこうしらは，　…………←キーで正しい文節の区切りを指定する
スペースバー→スペースバー→ ……
1.非行市　2.飛行士　3.非行　4.肥厚　5.披講　．．．．
スペースバーにより目的の語を選択する
水色反転
らは，
↓
ここで を入力してもよい。全体が一度にスクリーンカーソルの位置に移り確定する

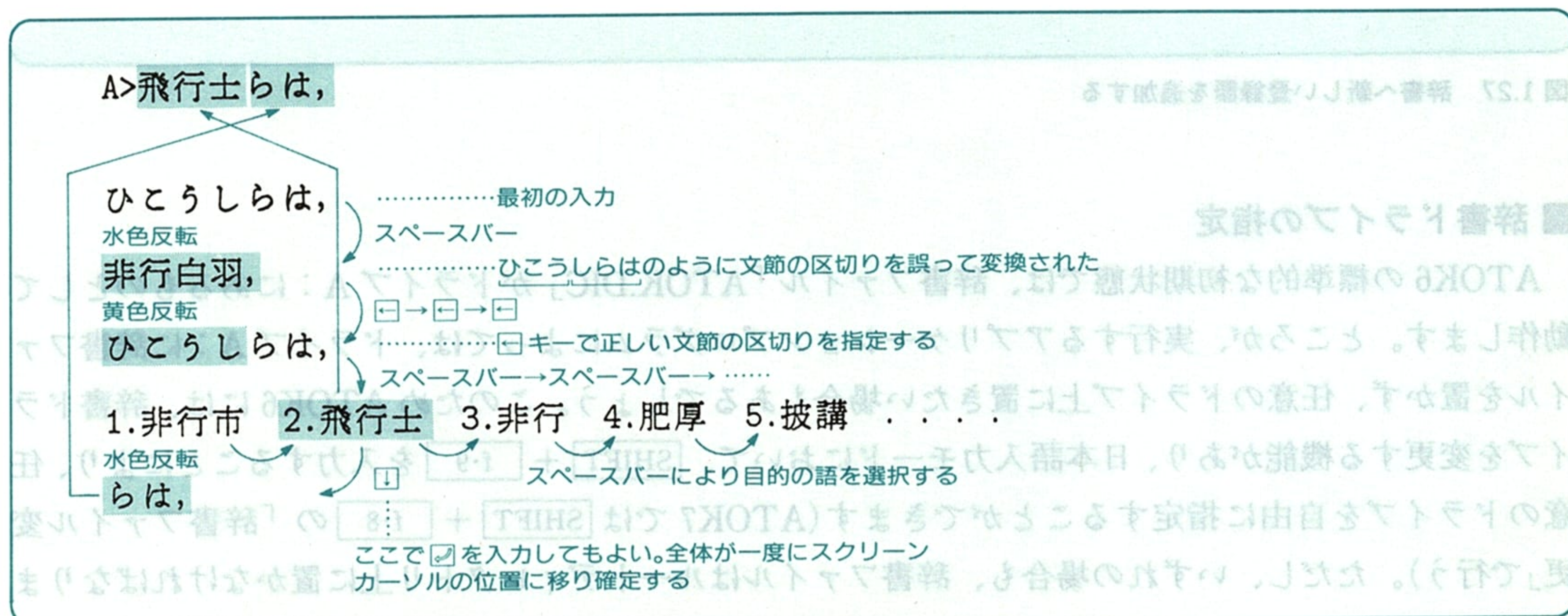



図 1.26b　ATOK6 が文節の区切り方を誤った場合の訂正法

■ 辞書登録語の追加／変更機能

ATOK6の辞書の登録語の追加や変更は、SHIFT ＋ f･5 （ATOK7では SHIFT ＋ f･6 ）によって行うことができます。では実行例として、「ワードプロセッサ」という単語を、「ワプ」という読みで登録してみましょう。

画面上の任意の位置にこの語が用意されているとする。
SHIFT＋f･5 を入力する

```
A>.....ワードプロセッサ.....

↑↓←→キーで始点を指定してください          単語登録　連ローマ字漢字
  (カーソルを「ワ」に合わせ⏎する)
←，→キーで終点を指定してください            単語登録　連ローマ字漢字
  (カーソルを「サ」に合わせ⏎する)
単語＝ワードプロセッサ　[Y,N] Y                       連ローマ字漢字
  (OKならばYを入力)
読みを入力してください                                 連ローマ字漢字
  (WAPUと入力し⏎)
わぷ_                                                  連ローマ字漢字
種類＝1 【 1.名詞　2.名詞(サ変) 3.固有名詞　4.その他】単語登録　連ローマ字漢字
  (番号を入力し⏎)
登録してよろしいですか？ [Y,N] Y              単語登録　連ローマ字漢字
単語登録終了                                  単語登録　連ローマ字漢字
_        登録完了                                      連ローマ字漢字
「わぷ」と入力して「ワードプロセッサ」に変換されるかどうかを実際に確認してみる
わぷ_ ……………キー入力                                連ローマ字漢字
  スペースバー
ワードプロセッサ ……………変換された！                連ローマ字漢字
```

図 1.27　辞書へ新しい登録語を追加する

■ 辞書ドライブの指定

ATOK6の標準的な初期状態では、辞書ファイル「ATOK.DIC」がドライブA：にあるものとして動作します。ところが、実行するアプリケーションプログラムによっては、ドライブA：に辞書ファイルを置かず、任意のドライブ上に置きたい場合もあるでしょう。このためATOK6には、辞書ドライブを変更する機能があり、日本語入力モードにおいて、SHIFT ＋ f･9 を入力することにより、任意のドライブを自由に指定することができます（ATOK7では SHIFT ＋ f･8 の「辞書ファイル変更」で行う）。ただし、いずれの場合も、辞書ファイルはルートディレクトリ上に置かなければなりません（ATOK7では、任意のディレクトリに置くことが可能）。

連ローマ字漢字

SHIFT＋f·1 の入力により、日本語入力ラインがこのように変わる

辞書ドライブ＝B　ファイル名＝ATOK.DIC　ファイル変更

変更したい任意のドライブ名をキー入力する

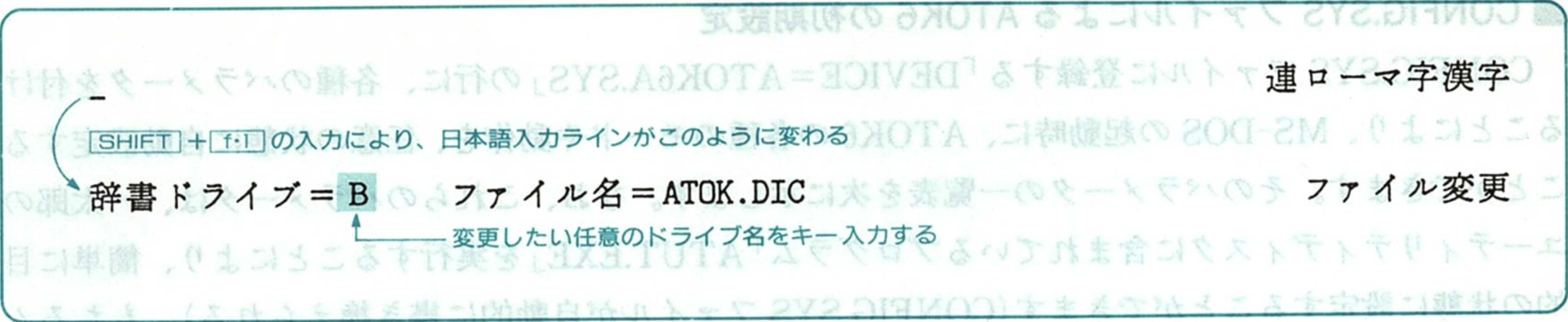



図 1.28　辞書ドライブの指定の変更

この操作以降は、ここで指定されたドライブ上の辞書ファイルが使われることになります。

■ 学習機能の ON／OFF

ATOK6 では、漢字などの登録語への変換の際、同じ読みの登録語が複数存在する場合には、もっとも最近に使われた登録語の順に取り出されます。これは自動学習機能（辞書ファイルの登録語の取り出し優先順位を書き変える機能）が働いているからですが、この学習機能は、必要があれば OFF にすることができます。日本語入力モードにおいて、SHIFT ＋ f·2 を入力することにより、自動学習機能が ON／OFF します（ATOK7 では、SHIFT ＋ f·10 で「学習モード」を選択して ON／OFF を設定する）。

この部分が反転している状態がON

連ローマ字漢字

SHIFT＋f·2 の入力により、辞書の学習機能をON⇄OFFできる

連ローマ字漢字

反転していない状態がOFF

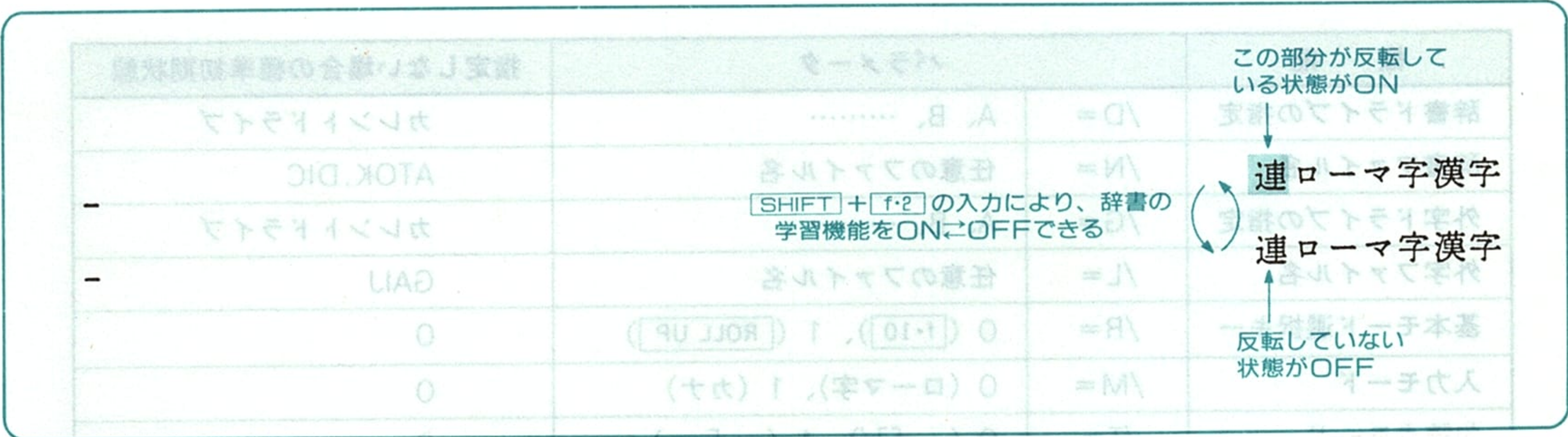



図 1.29　自動学習機能を ON／OFF する

■ CONFIG.SYS ファイルによる ATOK6 の初期設定

CONFIG.SYS ファイルに登録する「DEVICE＝ATOK6A.SYS」の行に、各種のパラメータを付けることにより、MS-DOS の起動時に、ATOK6 の各種のモードや動作を、任意の状態に自動設定することができます。そのパラメータの一覧表を次に示します。なお、これらのパラメータは、一太郎のユーティリティディスクに含まれているプログラム「ATUT.EXE」を実行することにより、簡単に目的の状態に設定することができます(CONFIG.SYS ファイルが自動的に書き換えられる)。もちろん「ATUT.EXE」を利用しなくても、エディタやワープロ、あるいは COPY コマンドなどを使って任意の内容の CONFIG.SYS ファイルを作成すればよいことです。

例：DEVICE＝ATOK6A.SYS_/D＝B_/N＝ATOKMY.DIC_/L＝GAIJMY_/E＝1_/C＝1

（「_」はスペースを表す）

- /C＝1：漢字コードにシフト JIS を指定
- /E＝1：スクリーンカーソル上で日本語入力を行う
- /L＝GAIJMY：外字ファイル名は「GAIJMY」
- /N＝ATOKMY.DIC：辞書ファイル名は「ATOKMY.DIC」
- /D＝B：辞書ドライブは B：

機　能	パラメータ		指定しない場合の標準初期状態
辞書ドライブの指定	/D＝	A、B、………	カレントドライブ
辞書ファイル名	/N＝	任意のファイル名	ATOK.DIC
外字ドライブの指定	/G＝	A、B、………	カレントドライブ
外字ファイル名	/L＝	任意のファイル名	GAIJ
基本モード選択キー	/R＝	0（[f・10]）、1（[ROLL UP]）	0
入力モード	/M＝	0（ローマ字）、1（カナ）	0
句読点モード	/T＝	0（,. []/)、1（、。「」・）	0
辞書学習機能	/S＝	0（しない）、1（する）	1
入力位置	/E＝	0（日本語入力ラインカーソル） 1（スクリーンカーソル）	0
文節変換モード	/B＝	0（連文節）、1（単文節）	0
漢字コードモード	/C＝	0（JIS）、 1（シフト JIS）、 2（区点）	0

表 1.2　ATOK6A の初期設定用の各種パラメーター覧表

1.3　新松に付属の日本語入力システム「松茸(まつたけ)」

「松」(管理工学研究所)は従来から「一太郎」と並び称されてきたワープロソフトであり、一太郎に対抗する一方の雄と言ってもよいでしょう。

松の歴史は、1983 年に発売された、PC-9801 の BASIC 上の「BASIC 版〝松〟」に始まります。これは MS-DOS に依存しない「スタンドアローン」(自分自身で動作する)のワープロソフトであり、BASIC ファイル形式の文書ファイルを出力しました。しかし、この「BASIC 版〝松〟」には、本章で取り上げている「FEP」という独立した機能は存在していませんでした。

ビジネスソフトにおける MS-DOS 採用の波は松にも及び、1986 年、MS-DOS 上に移植された「MS-DOS 版〝松〟」(松 86)が発売され、MS-DOS ファイル形式の文書ファイルの出力や FEP の独立など、使用環境が格段に改善された現在の松の原型が誕生しました。松はその後、「新松」にバージョンアップされ、今日に至っています。

松の FEP には、松 86 の時代から「松茸(まつたけ)」という名前が付けられています。松茸は、「ワープロ松」の FEP として、そのシステムディスクに組み込まれているものであり、松茸単独では販売されていません。しかしワープロ松からは、FEP の松茸の部分だけを取り出すことができ、取り出した松茸をユーザーの任意のディスクに組み込むことにより、ほかの多くのアプリケーションプログラム上で、松茸による日本語入力を利用することができます。

■ 松茸

ここでは PC-9800 シリーズ用の新松に付属の「松茸 V2」(バージョン 2)を紹介しましょう。松茸は、上で述べたように、ワープロソフト「新松」に使われている FEP です。松のバージョンアップとともに、松茸のバージョンも、松茸、松茸 86、松茸 V2 とアップされてきましたが、このいずれも、基本的な機能や操作法はほぼ同じです。

松茸を、私たちが使っている任意のアプリケーションプログラムで利用するには、新松の各種のプログラムファイルから、FEP である「松茸」に関する部分を取り出し、それをアプリケーションプログラムのディスクに組み込みます。この FEP の移植により、たいていのアプリケーションプログラムから、松茸の日本語入力機能を利用することができるようになります。このことを図解してみましょう。

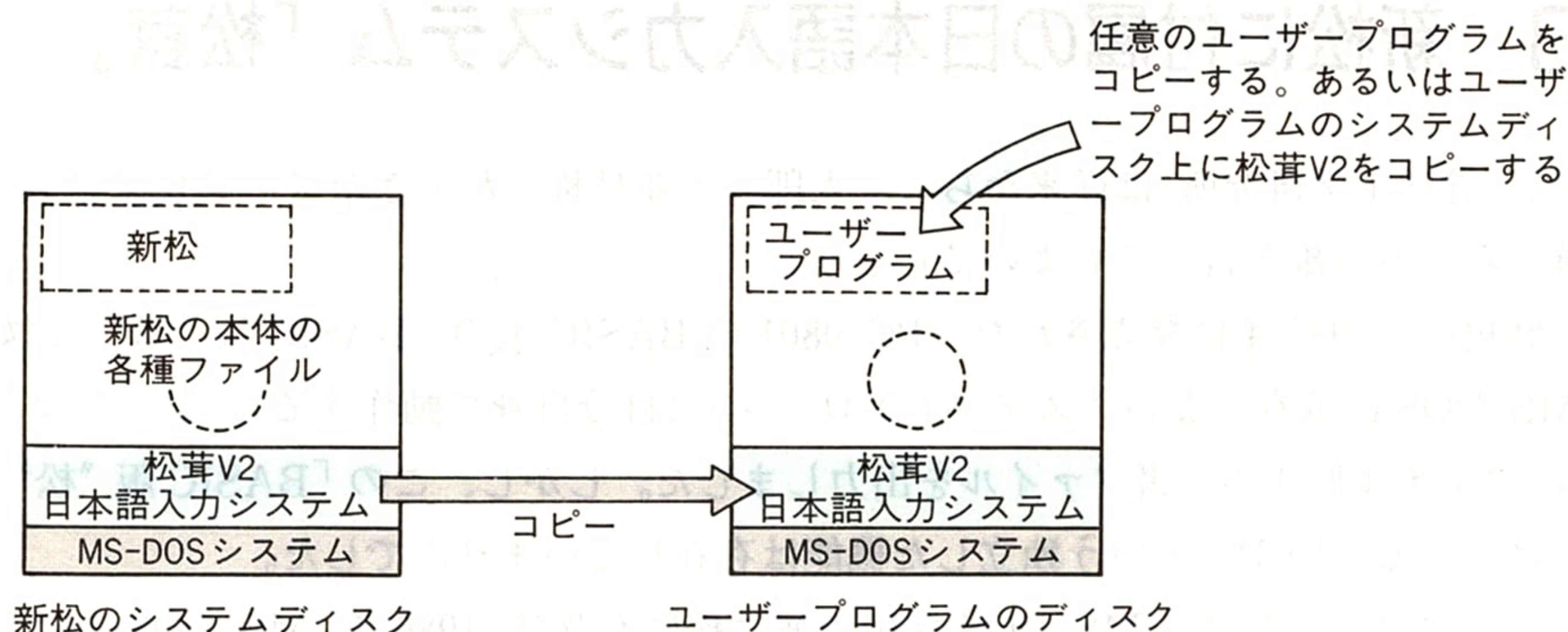



図 1.30　新松のディスクに含まれる FEP「松茸」とその移植

本章 1.1 で述べたように、松茸を始めとする FEP は、デバイスドライバとして、ワープロソフトの本体から独立していますので、この図のように、松茸に関する部分だけを取り出し、任意のアプリケーションプログラムのディスク上に組み込むことが可能なのです。本章では、FEP の機能（日本語入力機能）の紹介が目的ですので、新松の本体には言及せず、新松のシステムディスクから松茸のみを取り出して、松茸が機能する新しい MS-DOS システムディスクを作成し、それをもとに実習解説を行いましょう。

■ 松茸を組み込んだシステムディスクの作成

ユーザーが使用する各種のアプリケーションプログラム上で松茸の日本語入力機能を利用できるように、新松から松茸を移植する実例を示しましょう。さきにも述べたように、ここでは新松に付属の松茸 V2 を例にしますが、松 86 などの他のバージョンの松茸も、本書で解説する基本的な範囲ではほとんど同じです。また、松茸が動作するためには、松茸のみで約 60K バイトのメインメモリを占有しますので、考慮しておいてください。

ではまず、新松のシステムディスクに含まれている全ファイルのなかから、松茸（つまり FEP）に関連するファイルを図 1.31 に示します。

```
A>DIR ↵ ……………新松のシステムディスクに含まれるファイルの内容

 ドライブ A: のディスクのボリュームラベルは 松SYSTEM
 ディレクトリは A:¥

COMMAND  COM     17190  86-06-10    0:00
JISHO    DIC    481280  87-12-25    1:02 ┐
BUNPO    DIC     10114  87-12-25    1:02 │辞書ファイル(辞書本体/文法/索引の3つに分かれている)
INDEX    DIC      1280  87-12-25    1:02 ┘
GAIJI    DIC     19656  87-12-25    1:02 ……外字を使わない場合、外字辞書は必要ない
CONFIG   SYS        81  87-12-25    1:02 ……松茸V2をMS-DOSに組み込むためのCONFIGファイル
AUTOEXEC BAT        15  87-12-25    1:02
MTTK2    DRV     43920  87-12-25    1:02 ……松茸V2本体のプログラム(デバイスドライバ)
MOUSEK   DRV      4382  87-12-25    1:02
MATULOGO COM      1933  87-11-25    1:00
MATU     COM      9405  88-01-20    1:02
MATU     EXE    264352  88-01-20    1:02
MATU     SET     51282  88-03-03   21:53
MATU     HLP     58348  87-12-25    1:02
        14 個のファイルがあります.
    220160 バイトが使用可能です.

A>
```

図 1.31　新松のシステムディスクに含まれる松茸に関する各種ファイル

このリスト上の色でマークされたいくつかのファイルが、松茸の動作に必要なものです。これらのファイルを日常使用する MS-DOS システムディスクや、各種のアプリケーションプログラムのディスク上にコピーすればよいわけです。そのことを次ページの**図 1.32** で示しましょう。

では、この作業の実行例を示します。まず、図 1.32 の例 1(上側)に示されている手順により、フォーマット処理した空ディスクに MS-DOS システムをコピーし、その上に松茸の各種のファイルをコピーして、松茸を組み込んだ新しい MS-DOS システムディスクを作成します。

MS-DOS システム(COMMAND.COM を含む)をコピーするには、「/S」付き FORMAT コマンドや SYS コマンドを使います。コピー元にする MS-DOS システムには、MS-DOS システムディスクや、新松の常用ディスク(日常の作業に使っているディスク)上の MS-DOS システムが使えます。

新しい MS-DOS システムディスクが作成できたら、図 1.31 に示した松茸関係の各種のファイルを COPY コマンドでコピーすれば、ひとまず作業は終わりです。

〈松茸V2の組み込み〉

■ 例1.新しいMS-DOSシステムディスクを作成する場合

新松
松茸V2
日本語入力システム
MS-DOSシステム
新松オリジナルディスク

松茸V2に関する最低限必要なファイル
JISHO.DIC
BUNPO.DIC
INDEX.DIC
MTTK2.DRV
CONFIG.SYS

松茸V2の各ファイルをコピー

MS-DOS
システムディスク
MS-DOSシステム
MS-DOSシステムディスク

どちらのMS-DOS
システムをコピー
してもよい

ここに実行したい任意の
ユーザープログラムをコ
ピーして松茸V2を利用する

松茸V2各ファイル
MS-DOSシステム
松茸V2が組み込まれた新し
いMS-DOSシステムディスク
ができあがる

■ 例2.すでに使用中のユーザーディスク上にコピーする場合

新松
松茸V2
日本語入力システム
MS-DOSシステム
新松オリジナルディスク

松茸V2の各ファイルをコピー

ユーザー
プログラム
松茸V2各ファイル
ユーザープログラム
MS-DOSシステム
すでに使用中の、ユーザー
プログラムの実行用ディス
ク上に、松茸V2をコピーし
て利用する

図 1.32　松茸を移植する２つの方法

では、COMMAND.COM を含んだ MS-DOS システムディスクに、松茸関係の各種のファイルをコピーする実行例を図 1.33 に示しましょう。

```
A>FORMAT B:/S ⏎ ………ドライブB：のディスクをフォーマットし、ドライブA：のMS-DOSのシステムをコピーする
Format Version 4.10

      ⋮（途中省略）

別のディスクをフォーマットしますか(Y/N) N ⏎ ……………FORMATコマンドを終了する
A>COPY *.DIC B: ⏎
JISHO.DIC
BUNPO.DIC    ワイルドカードによってこれらの
INDEX.DIC    ファイルがコピーされる
GAIJI.DIC
        4 個のファイルをコピーしました.
                                              ドライブA：の新松のシステムディスク上のこれ
A>DEL B:GAIJI.DIC ⏎ ………このファイルは必要ないので削除   らのファイルを、ドライブB：にコピーする。

A>COPY MTTK2.DRV B: ⏎
        1 個のファイルをコピーしました.

A>COPY CONFIG.SYS B: ⏎
        1 個のファイルをコピーしました.

A>DIR B: ⏎ ……………結果の確認

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

COMMAND  COM     24931  88-10-16  16:10 ……………「FORMAT /S」コマンドでコピーされたもの
                                                  (MS-DOSシステムもコピーされている)
JISHO    DIC    481280  87-12-25   1:02
BUNPO    DIC     10114  87-12-25   1:02
INDEX    DIC      1280  87-12-25   1:02   COPYコマンドでコピーされたもの
MTTK2    DRV     43920  87-12-25   1:02
CONFIG   SYS        81  87-12-25   1:02
        6 個のファイルがあります.
   590848 バイトが使用可能です.

A>
```

図 1.33　松茸を組み込んだ MS-DOS システムディスクの作成

以上の作業で、松茸に関するすべてのファイルがコピーされた、MS-DOS システムディスクがドライブ B：上にできあがりました。松茸による日本語入力が行える MS-DOS システムディスクが用意できたわけです。

このシステムディスクをドライブ A：にセットしてリセットボタンを押せば、MS-DOS が起動し、同時に松茸も動作可能な状態になるはずですが、その前に **CONFIG.SYS ファイル**（コンフィギュレーションファイル）の内容を確認してください。この内容によっては、図 1.31 に示した各種のファイルがコピーされていても、松茸は動作しません。では、図 1.33 でコピーした CONFIG.SYS ファイルの内容（新松のオリジナルディスクに付属しているもの）をタイプアウトしてみましょう。

```
A>TYPE CONFIG.SYS↵ ……………新松のオリジナルディスクに付属のCONFIG.SYSファイルの内容を
                                                                タイプアウトする
BUFFERS= 15
FILES  = 12          ┌──この部分は絶対に必要
DEVICE = MTTK2.DRV /N /T /G /J4     この部分は松茸V2の初期入力モードを設定するための
DEVICE = MOUSEK.DRV                 環境パラメータであるが、当面は省略してよい。パラメ
                                    ータを省略した場合は、標準状態となる。くわしくは後
                                    述。またユーティリティプログラム「MINSTALL.
A>         └──この行は必要ないので削除する   EXE」(新松の補助ディスクに含まれている)を実行す
              図1.35を参照              れば、これら各種のオプションが自動的に設定される
```

図 1.34　MS-DOS の起動時に、松茸を組み込むために必要な CONFIG.SYS ファイルの内容

ここで示されている CONFIG.SYS ファイルの内容の中で、

DEVICE＝MTTK2.DRV

の部分は、松茸を MS-DOS に組み込むために絶対に必要です。各自のディスク上の CONFIG.SYS ファイルをタイプアウトして確認してください。この部分が記述されていない場合は、エディタやワープロを使ったり、後述の COPY コマンドなどで CONFIG.SYS ファイルの内容を書き換えなければなりません。CONFIG.SYS ファイルの機能については、本章の 1.1 や、3.1 章で触れていますが、そのファイルの内容で指示される、各種のデバイスドライバの MS-DOS システムへの組み込みや、環境の設定が、MS-DOS の起動時に実行されるものと考えればよいでしょう。

CONFIG.SYS ファイルは、テキストファイル(文字ファイル)です。これを編集するには、エディタやワープロを利用しますが、CONFIG.SYS ファイルのような、短い簡単な内容であれば、MS-DOS の COPY コマンドを使って、新たに作成してもかまいません。その実行例を示します。次のような内容の CONFIG.SYS ファイルを作成すればよいでしょう。

```
A>COPY CON CONFIG.SYS↵ ……………COPYコマンドを使って、CONFIG.SYSファイルを作成する
FILES=10↵    ┐
BUFFERS=20↵  ┘この2行はなくても松茸V2の動作は可能(3.1章を参照)
DEVICE=MTTK2.DRV↵ ……………この行は絶対に必要
^Z ↵ ……………CTRL+Z を入力した後、↵する
        1 個のファイルをコピーしました. ……………この時点で、キー入力された文字の内容の
                                        ファイルが作成されMS-DOSにもどる
A>TYPE CONFIG.SYS↵ ……………作成されたファイル「CONFIG.SYS」をタイプアウトして確認する
FILES=10          ┐
BUFFERS=20        ├このように作成されている
DEVICE=MTTK2.DRV  ┘

A>
```

図 1.35　松茸を動作させるための CONFIG.SYS ファイルを作成する

さて、COPY コマンドを使って、新しい CONFIG.SYS ファイルが作成されたとしましょう。CONFIG.SYS ファイルの内容を確認し、上の行が含まれていれば松茸による日本語入力を行うことができます。ここで作成したディスクには、MS-DOS の本体と松茸が含まれているだけですが、各種の実務には、それぞれのアプリケーションプログラムや、必要なファイルをこのディスク上にコピーした上で使用することになります。

アプリケーションプログラムの常用ディスクに松茸を組み込む

次に、44 ページの図 1.32 の例 2(下側)の方法で、松茸の各ファイルを目的のディスクにコピーする実行例を示しましょう。具体的には、45 ページの図 1.33 の実行例に示した COPY コマンドの部分を、目的のアプリケーションプログラムの常用ディスクに対して実行し、松茸の各ファイルをコピーすればよいのです。もしそのディスクに、すでにほかの FEP が存在している場合は、ディスクの容量を考慮して、事前にその FEP に関するファイルを削除しておきます(ハードディスクなどで、容量に十分な余裕があればその必要はない)。とくに辞書ファイルは大容量であるため、フロッピーディスクの場合は、2 種類の FEP を同時に収容しておくことはできません。

この場合も、CONFIG.SYS ファイルの内容は重要です。もし、すでに CONFIG.SYS ファイルが存在していれば、ほかの FEP に対応した内容であるかもしれません。その場合は、エディタやワープロで内容を書き換えるか、追加するか、作成し直さなければなりません。前項を参照して、CONFIG.SYS ファイルの内容を松茸用に変更してください。

■ 松茸による日本語入力の操作法

では、例 1 や例 2 の方法で作成したシステムディスクをドライブ A：にセットし、リセットボタンを押して、MS-DOS を再起動してみましょう。松茸のシステムは、MS-DOS の起動時に、デバイスドライバとしてメモリ上の MS-DOS システムに組み込まれます。**重要！**

```
(リセットボタンを押す)
                    (松茸V2は、約60Kバイトのメモリを消費します)
NEC PC-9800 Series Personal Computer

ﾏｲｸﾛｿﾌﾄ MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30
Copyright 1981,88 Microsoft Corp. / NEC Corporation
--------------------------------------------------------------------
日本語入力フロントプロセッサ「松茸」Ver 2.01 (C) 1987 (株)管理工学研究所
--------------------------------------------------------------------

Command ﾊﾞｰｼﾞｮﾝ 3.30
現在の日付は 1989-08-10 (木) です.
日付を入力してください:
現在の時刻は 19:37:11.00 です.
時刻を入力してください:

A> ……………MS-DOSが完全に起動した
```

松茸V2日本語入力システムが、MS-DOSシステムに組み込まれたことを示すオープニングメッセージ

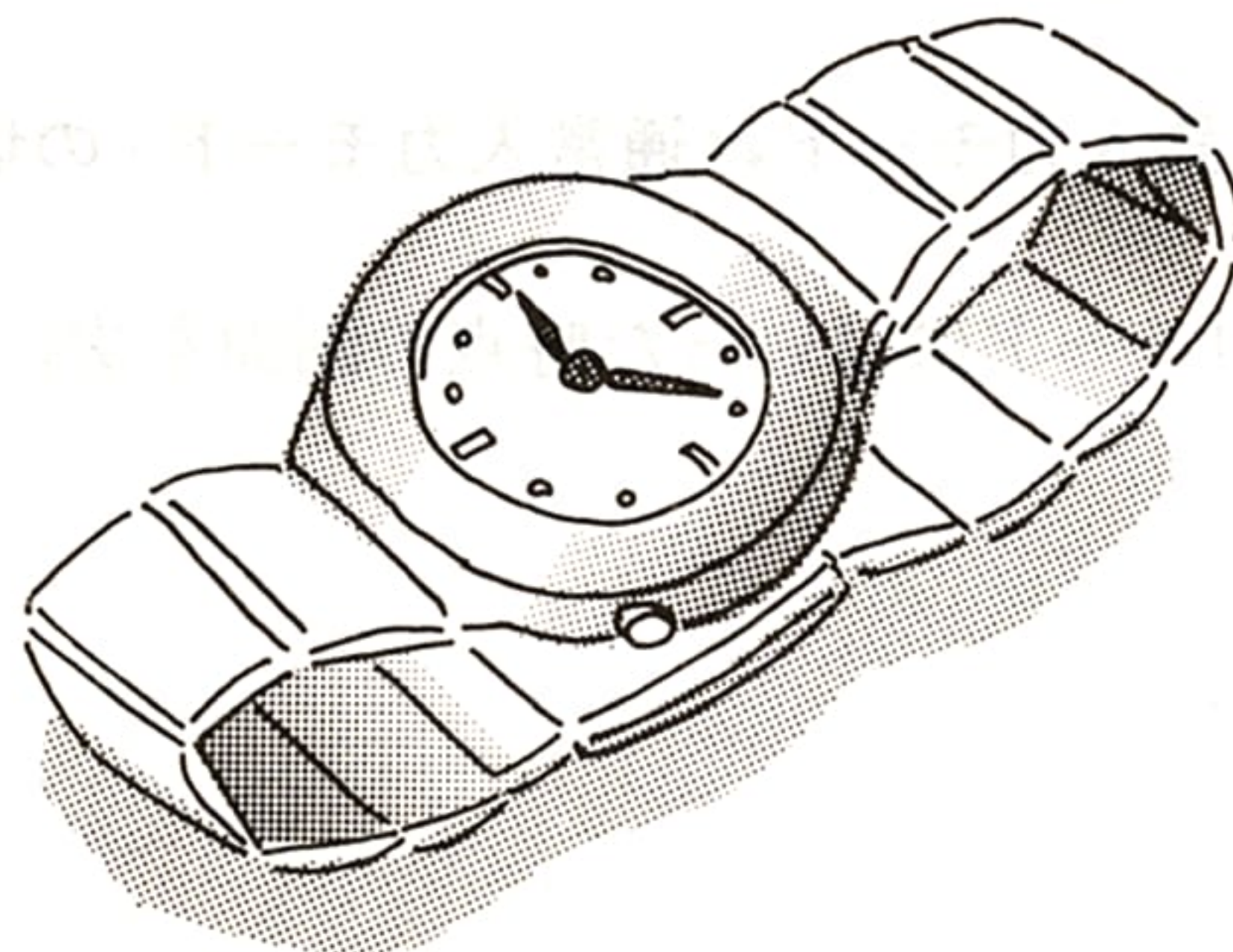
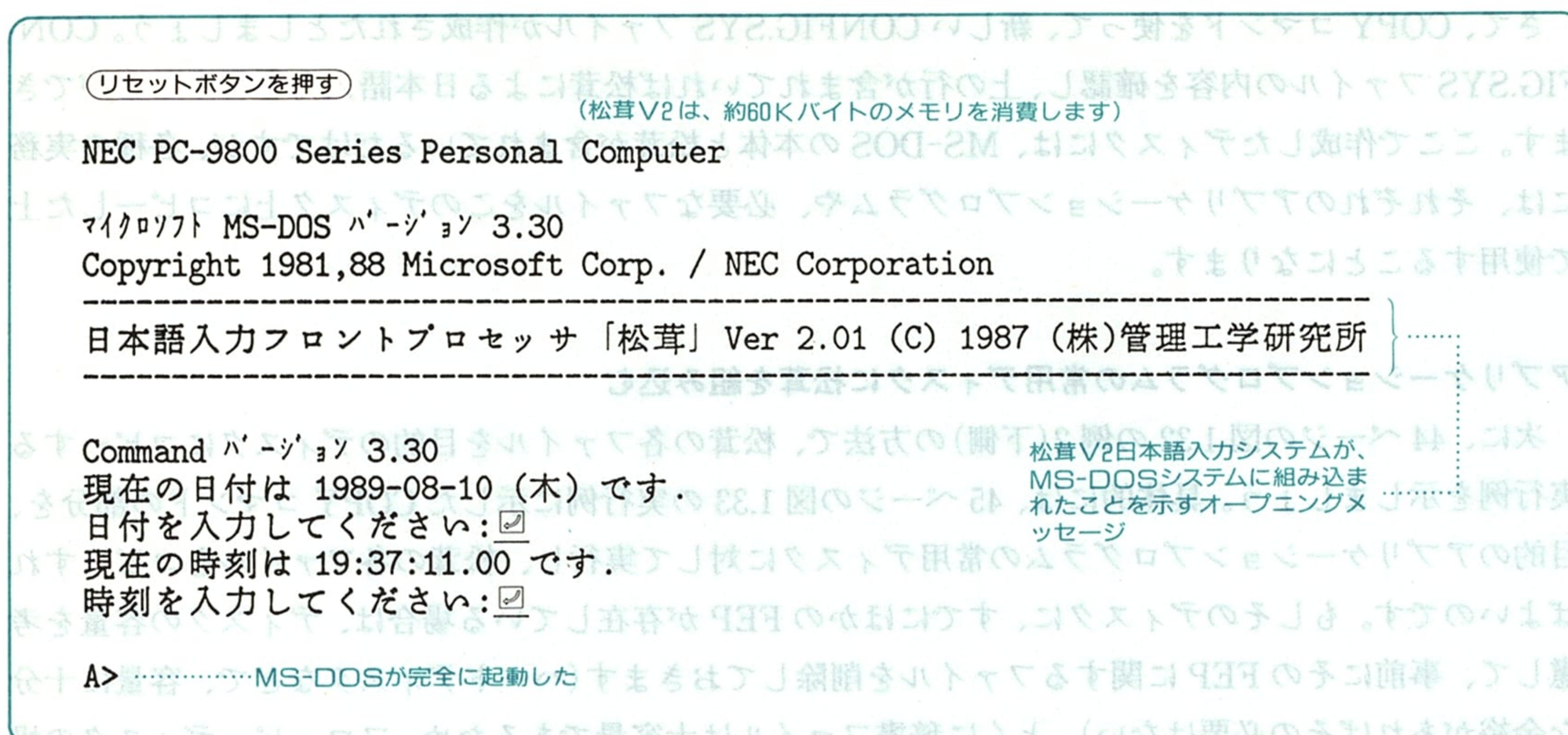



図 1.36　松茸を組み込んだ MS-DOS システムディスクの起動

MS-DOS のオープニングメッセージと、松茸が組み込まれたことを示すメッセージが表示され、MS-DOS が立ち上がりました。これで松茸による日本語入力が可能になりました。もし、前項(図 1.32 の例 2 の場合)で作成したディスクのアプリケーションソフトが、自動的に立ち上がってしまった場合は、そのソフトを「終了」して MS-DOS におり、プロンプト「A>」の状態にしてください。

さて、松茸が動作する環境が整ったとして、次は実際に日本語入力を行うための操作の基本を実習しましょう。前節でも述べたように、松茸による日本語入力も他の FEP の場合と同様、プロのタイピストを志望する人を除き、ローマ字入力で行うことを強くお勧めします。

まず、FEP が機能する状態と、機能しない状態との切り替え操作です(FEP の ON／OFF 操作)。日本語入力が可能な状態(**日本語入力モード**)と、日本語入力機能が働かない状態(**通常入力モード**)との切り替えは、[XFER] キー、あるいは [CTRL] ＋ [XFER] ([CTRL] キーを押しながら [XFER] キーを押す)で行います。日本語入力モードから、通常入力モードにもどるには、再度 [XFER] キー、あるいは [CTRL] ＋ [XFER] を入力します。

「日本語入力モード⇄通常入力モード」の切り替えは、[XFER]、あるいは [CTRL] ＋ [XFER] で行う

日本語入力モードにはいった時点の画面を次に示します。

A>█ ← スクリーンカーソル。直接入力なので、この場でかな－漢字変換を行う

Rか　A↓　貼込　8:19　記号　複写　松茸ｶﾅ変換　半角変　ﾛｰﾏ字　半角　カナ　Rか

現在の入力モードの表示(ローマ字入力モード)　　左端と同じ入力モードが表示される

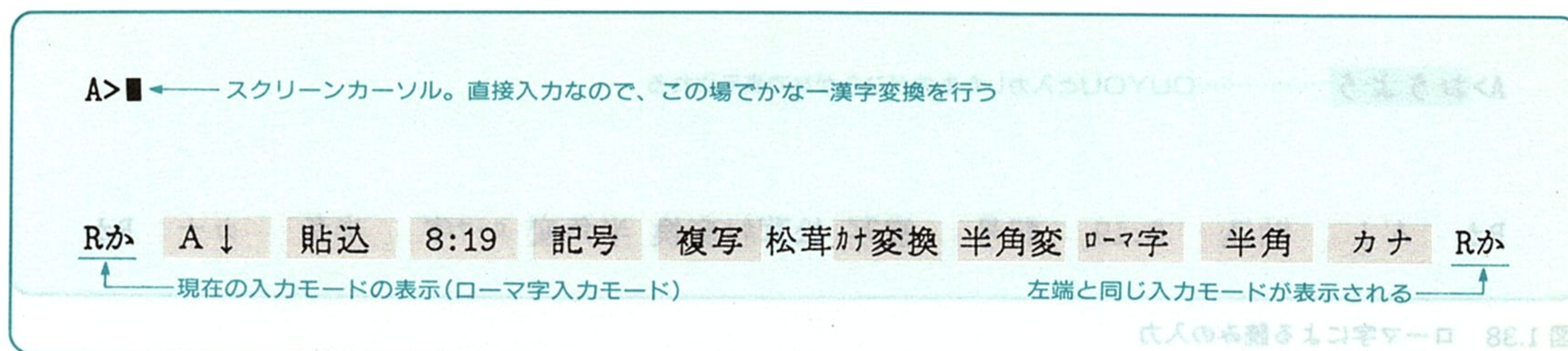



図 1.37　松茸の日本語入力モードにはいった初期画面

まず松茸の日本語入力モードには、

- **間接入力モード**
- **直接入力モード**

の2つがあります(この選択は、CONFIG.SYS ファイル内の記述で行う。後述)。

間接入力モードというのは、画面の最下段に日本語入力ラインが設けられ、このライン上で、「読み」を入力したり、かな－漢字変換をしたりして、目的とする文字や文字列を準備してから、スクリーンカーソルの位置へ移す入力形式です。ATOK の標準モードと同様の形式です。

直接入力というのは、日本語入力ラインを設けずに、「読み」の入力も、かな－漢字変換も、スクリーンカーソルの位置で直接行う入力形式です。どちらの形式を使うかは、松茸を利用するアプリケーションプログラムの都合によって選択します。一般的には直接入力モードの方が使いやすいのですが、アプリケーションプログラムによっては、間接入力モードでなければうまく入力できない場合があるかもしれません。ここでの実習は、画面下部にファンクションメニューが表示されるなど、学習するためにも都合のよい、直接入力モードを主体に解説しますが、間接入力モードの場合も基本的な操作は同じです。

さて、画面最下段のメニューラインの左右端に、**Rか**と表示されているように、松茸では、日本語入力モードにはいった時点の標準設定は、ローマ字入力モードになっています(CONFIG.SYS ファイルの記述により任意の初期モードに設定可能。後述)。ではこの状態で、とりあえず「応用 MS-DOS」という文字列を入力してみましょう。

[カナ] キーが OFF であることを確認した後、「OUYOU」とキー入力します([カナ] キーが OFF でなければローマ字(英字)入力はできない)。英字を入力する際、大文字／小文字を選択する [CAPS] キーは、ON でも OFF でもどちらでもかまいません。英字そのものを入力する際に、大文字／小文字のどちらの頻度が多いかによって、使いやすい方を選択しておけばよいでしょう。ミスタイプした場合は、[BS] でカーソルの左の1文字を、[DEL] でカーソル上の1文字を削除することができます。

A>おうよう …………OUYOUと入力したものがひらがなで表示される

Rか　A↓　貼込　8:19　記号　複写　松茸ｶﾅ変換　半角変　ﾛｰﾏ字　半角　カナ　Rか

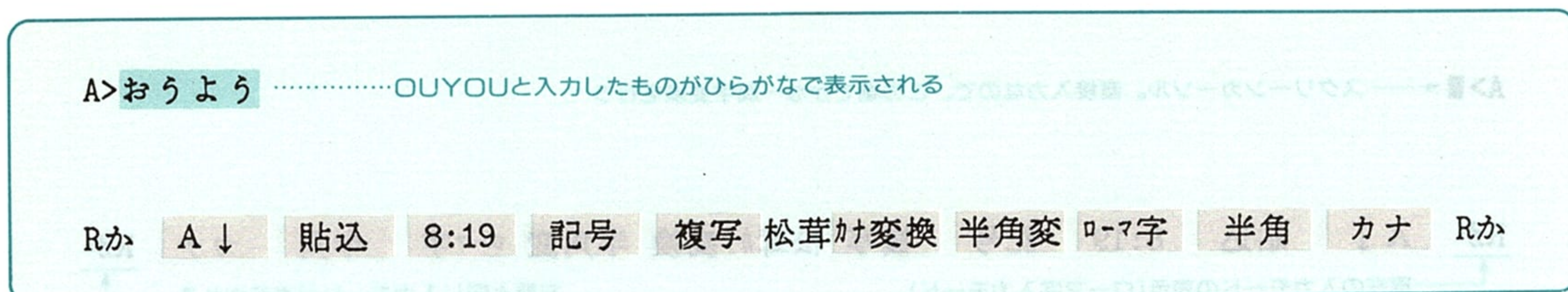



図 1.38　ローマ字による読みの入力

このようにローマ字で入力したものが、その場で自動的にひらがなに変換されながら、スクリーンカーソルの位置に「おうよう」と直接入力されます(日本語入力ラインを介して間接入力する場合は、CONFIG.SYS ファイルの「DEVICE=¥MTTK2.DRV」の記述にパラメータ「/I」の指定が必要。後述)。この読みを漢字(漢字だけとは限らないので、以後は**登録語**と呼ぶことにする)に変換するには、変換キーの［XFER］を入力します。

A>応用 …………変換キー［XFER］の入力で登録語に変換される

Rか　A↓　貼込　8:19　記号　複写　松茸ｶﾅ変換　半角変　ﾛｰﾏ字　半角　カナ　Rか

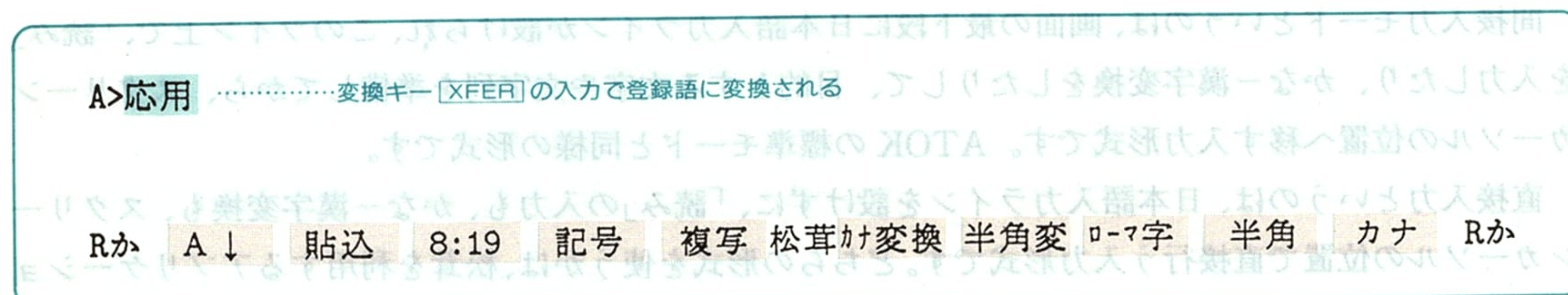



図 1.39　変換キー［XFER］の入力により漢字(登録語)に変換される

このように辞書ファイルに登録されている語のなかから、「応用」が取り出されています。この場合はこれで正解ですが、目的とする登録語に変換されなかった場合は、さらに［XFER］を入力することにより、同じ読みの別の登録語(次の候補)を取り出すことができます(もちろん登録されていればの話ですが)。あるいはその状態で、カーソル移動キーの［↓］を入力することにより、複数の次候補群が画面下のメニューラインに表示されますので、そこに目的の語があれば、その番号、あるいは［↓］や［↑］で反転カーソルを移動し［↵］することにより選択します。なければどんどん［↓］を入力して、さらに次の候補群のなかから捜します。また、以前に出てきた候補群にもどしたい場合には、［↑］をその分だけ入力します。

目的の文字列への変換ができたなら、続けて次の語の読みを入力するか、あるいは［↵］を入力することにより、それまで入力した文字列が確定されます(ここでの実行例におけるスクリーンカーソルは、MS-DOS のプロンプト「A>」の位置ですが、実際には各種のビジネスソフトや、エディタなどの入力画面のカーソル上であるわけです)。

「応用」と入力できたら、次は「MS-DOS」とキー入力します。この場合、そのままの状態で［M］［S］［-］［D］［O］［S］と入力すると、「MS—どS」というぐあいにローマ字変換されて入力されますので、英

語の綴りそのものの入力(アルファベットの入力)には、[f·8]を入力して「英数字入力モード」にしてから入力します。再度の[f·8]の入力で「ローマ字モード」にもどります。確定はさきほどと同様、次の語の読みを入力するか、あるいは[↵]を入力します。

以上の操作によって、目的の語「応用MS-DOS」を入力することができました。その画面を次に示します。

A>応用ＭＳ－ＤＯＳ …………[f·8]を入力して「英数字入力モード」にした後、MS-DOSと入力する

Rか　A↓　貼込　8:19　記号　複写　松茸ｶﾅ変換　半角変　ﾛｰﾏ字　半角　カナ　Rか

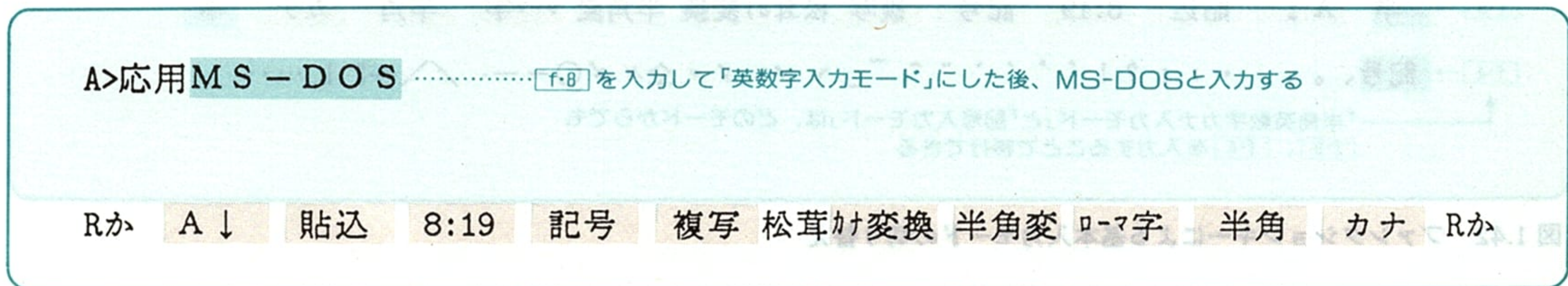



図 1.40　「英数字入力モード」により「MS-DOS」を入力する

入力モードの選択

松茸のメニューライン(ファンクションメニュー)は次に示す「A」「B」の2種類があり、[f·1]の入力により交互に切り替わります。

現在の時刻が表示される

f·1　f·2　f·3　f·4　f·5　f·6　f·7　f·8　f·9　f·10

(メニューA)…　A↓　貼込　8:19　記号　複写　松茸ｶﾅ変換　半角変　ﾛｰﾏ字　半角　カナ

[f·1]キーでメニューAとメニューBが交互に切り替わる

(メニューB)…　B↓　ﾌﾘｶﾞﾅ　ｺｰﾄﾞ表　ＪＩＳ　辞書　先読　学習　ﾄﾞﾗｲﾌﾞ　9-05　電卓　モード

現在の日付が表示される

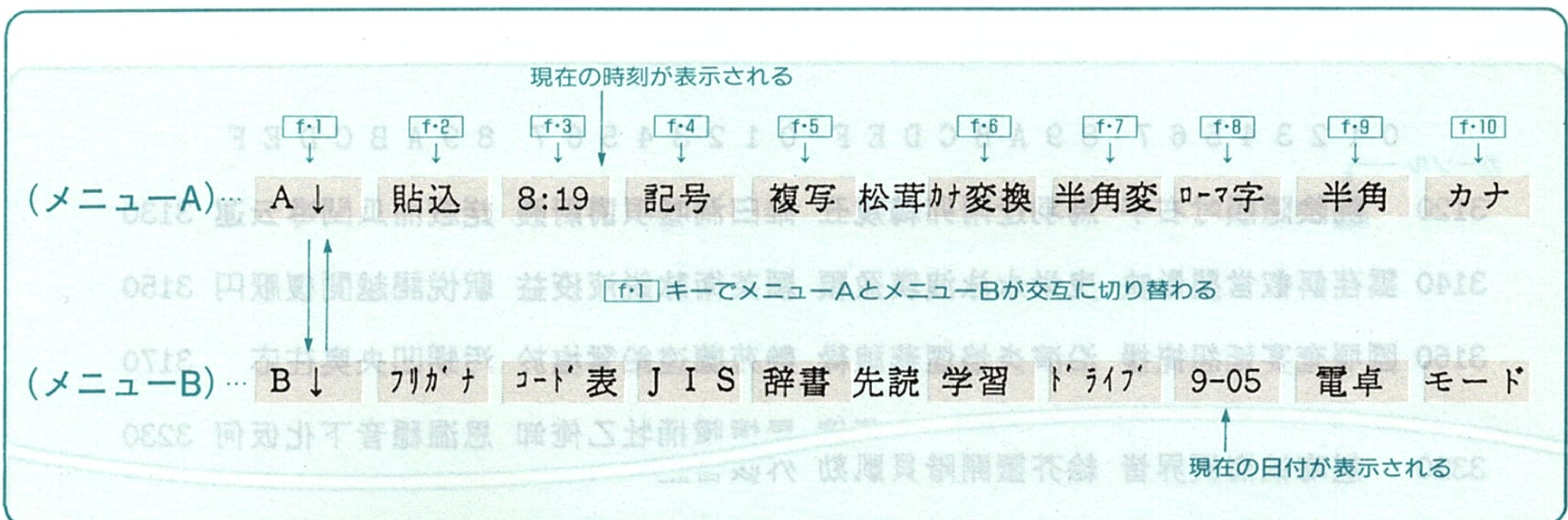



図 1.41　松茸のファンクションメニュー

このファンクションメニューA／Bの表示は、それぞれファンクションキーの[f·1]～[f·10]に対応し、メニューの表示に該当するファンクションキーを押すことにより、それぞれの機能が働きます。

松茸には、4つの基本的な入力モードがあり、それぞれのファンクションキーによって選択します。選択されているモードは、メニューラインの左右端の表示によって知ることができます。その4つの入力モードの切り替えの状況を次ページの図 1.42 に示します。

Rか　A↓　貼込　8:19　記号　複写　松茸カナ変換　半角変　ローマ字　半角　カナ　Rか
Rカ　A↓　貼込　8:19　記号　複写　松茸カナ変換　半角変　ローマ字　半角　カナ　Rカ
か　A↓　貼込　8:19　記号　複写　松茸カナ変換　半角変　ローマ字　半角　カナ　か
カ　A↓　貼込　8:19　記号　複写　松茸カナ変換　半角変　ローマ字　半角　カナ　カ
f·9 → 半　A↓　貼込　8:19　記号　複写　松茸カナ変換　半角変　ローマ字　半角　カナ　半
f·4 → 記号、。，．・：；？！゛゜´｀¨＾￣＿ヽヾゝゞ〃仝々〆〇ー―‐／＼～‖｜…‥

f·8　f·10　f·10　f·8

「半角英数字カナ入力モード」と「記号入力モード」は、どのモードからでも
f·9、f·4 を入力することで移行できる

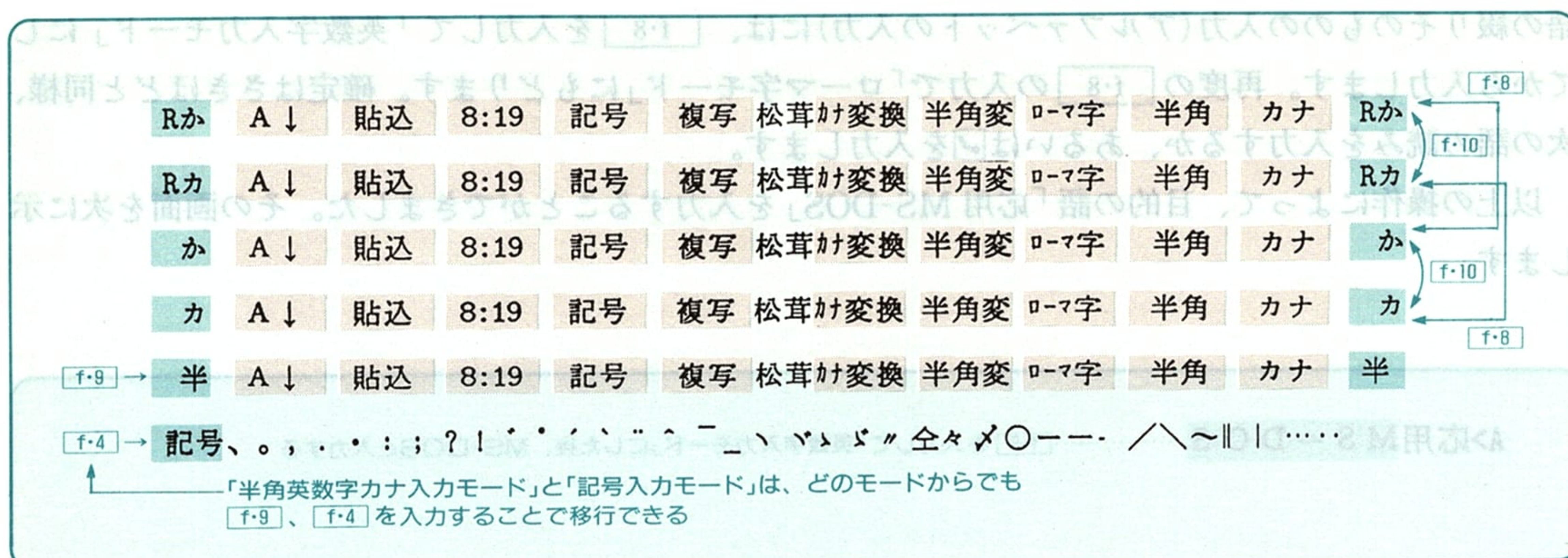



図 1.42　ファンクションキーによる基本入力モードの切り替え

これらの基本入力モードのうち、メニューB［f·3］の「コード表」、およびメニューB［f·4］の「JIS」の各入力モードは、JIS 漢字コードによって目的の文字(単字)を選択するもので、漢字コードは「JIS コード」を使います。なお、「コード表」モードの場合は、JIS コード表が画面全体に順に表示されますので、JIS コードそのものを入力することなく、［ROLL UP］、［ROLL DOWN］や、カーソル移動キーを使って目的の字を選択し⏎します。また、ファンクションメニューにもどるには、［ESC］を入力します。

では、これらのモードにおける JIS 漢字コードによる単字変換の実行例を示しましょう。

　　 0 1 2 3 4 5 6 7　8 9 A B C D E F　0 1 2 3 4 5 6 7　8 9 A B C D E F
カーソル
3120　 院陰隠韻吋右宇　烏羽迂雨卯鵜窺丑　碓臼渦嘘唄欝蔚鰻　姥厩浦瓜閏噂云運　3130
3140　雲荏餌叡営嬰影映　曳栄永泳洩瑛盈穎　頴英衛詠鋭液疫益　駅悦謁越閲榎厭円　3150
3160　園堰奄宴延怨掩援　沿演炎焔煙燕猿縁　艶苑薗遠鉛鴛塩於　汚甥凹央奥往応　3170
桶牡乙俺卸　恩温穏音下化仮何　3230
3320　 魁晦械海灰界皆　絵芥蟹開階貝凱劾　外
3340　垣柿蛎鈎劃嚇各廓　拡撹格核殻獲確穫　覚角赫較郭閣隔革　学岳楽額顎掛笠樫　3350
3360　橿梶鰍潟割喝恰括　活渇滑葛褐轄且鰹　叶椛樺鞄株兜竃蒲　釜鎌噛鴨栢茅萱　3370
JISコード　JISコード
↑↓→← キーでカーソルを移動
次画面へ移動　前画面へ移動　漢字を選択
ROLL UP:次　ROLL DOWN:前　カーソル:移動　RET:選択　ESC:終り　松茸 バージョン 2.01

図 1.43　JIS 漢字コードによる単字変換

また、メニューA [f･4] の「記号」モードは、さきのメニューB [f･3] の「コード表」モードと似ていますが、表示される文字が、記号関係の文字に限られています。いずれもカーソル移動キーを使って目的の字を選択し⏎します。

いずれにしても、ファンクションメニューAの左右端に、「Rか」(ローマ字による入力モードの場合)、または「か」(カタカナキーによる入力モードの場合)と表示されている状態が、松茸で日本語入力を行う場合の基本状態です。いろいろな状態から、この状態へのセットの仕方をよく覚えておいてください。

各種変換機能の基本操作

ここで、松茸の主要なキー操作と、注意しなければならないローマ字入力の綴り方などを表にまとめて示します。

■入力モードの切り替え

[XFER]
[SHIFT]＋[XFER]　} 通常入力モード⇄日本語入力モードの切り替え
A [f･4] (記号)……各種記号の入力
A [f･9] (半角)……全角入力モード⇄半角入力モードの切り替え
A [f･10] (カナ)……ひらがな入力モード⇄カタカナ入力モードの切り替え

■変換

[XFER] ……………………連文節変換
[SHIFT]＋[XFER] ………複合語変換
[CTRL]＋[XFER] ………単漢字変換
A [f･6] (カナ変換)……ひらがな⇄カタカナ変換
A [f･7] (半角変)………全角⇄半角変換
A [f･8] (ローマ字)……ローマ字かな変換⇄無変換

■学習機能 ON/OFF、辞書ドライブの変更、モードの切り替え

B [f･5] (辞書) ………………………ユーザー辞書登録／削除
B [f･6] (学習) ………………………辞書学習機能の ON/OFF
B [f･7] (ドライブ) …………………辞書ファイルのドライブやパス名の変更
B [f･10] (モード)
- 1.複合語 ……………文節変換／複合語変換の優先順位の切り替え
- 2.句読点 ……………「、。」または「，．」の切り替え
- 3.色 …………………未確定文字の色の選択
- 4.間接 ………………直接入力モード／間接入力モードの切り替え
- 5.先読み ……………辞書を先読みするかどうかの切り替え
- 6.半角入力 …………半角モードでの確定／未確定の切り替え
- 1.時計表示 …………松茸が OFF の時、時計を表示するかどうかの切り替え
- 2.句読点変換 ………句読点で変換するかどうかの切り替え

図 1.44a　松茸における各種キーの主要な機能

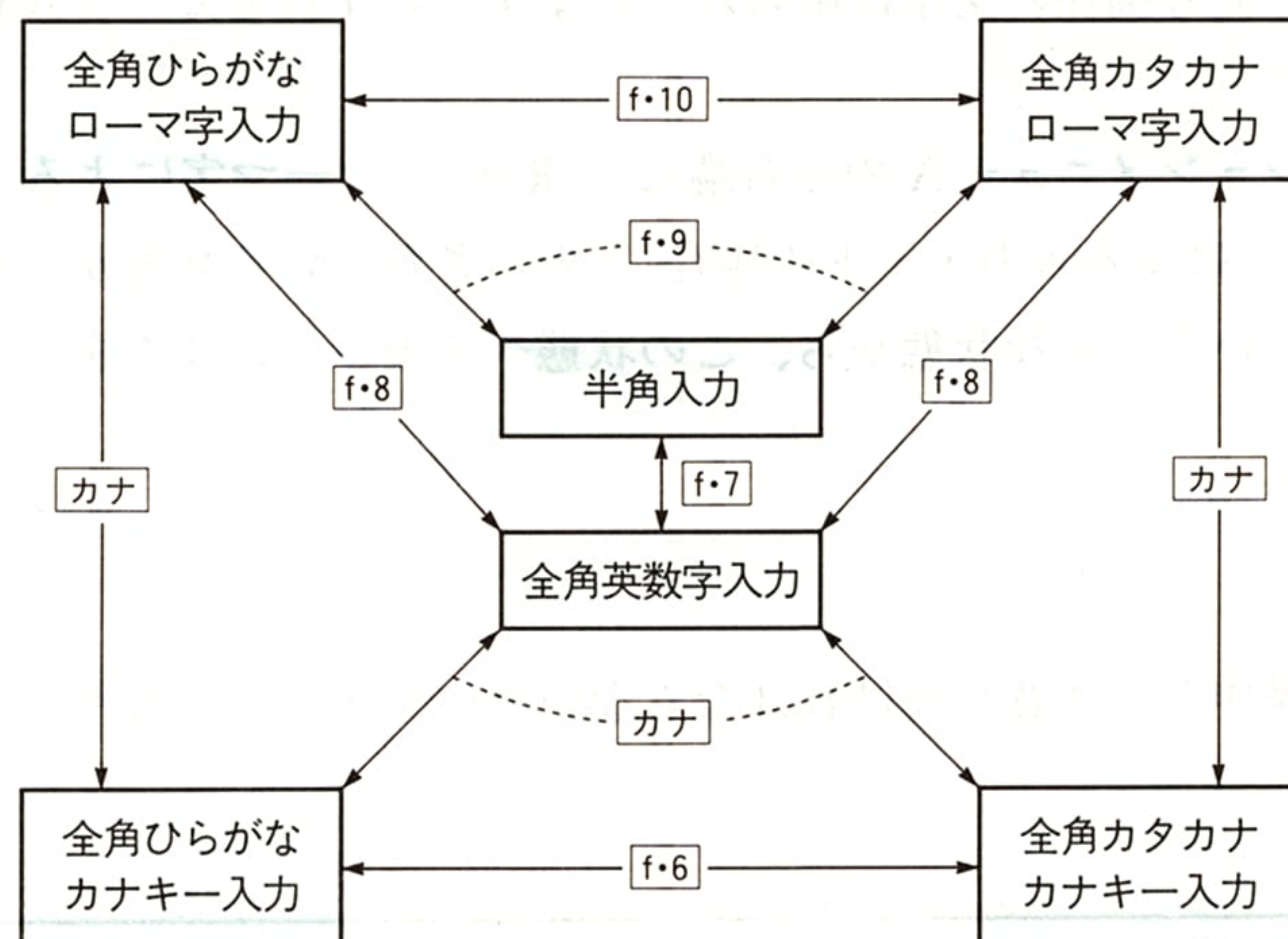
全角ひらがな ローマ字入力
f・10
全角カタカナ ローマ字入力
f・9
f・8
半角入力
f・8
カナ
f・7
カナ
全角英数字入力
カナ
全角ひらがな カナキー入力
f・6
全角カタカナ カナキー入力

これらの状態の切り替えは、すべてファンクションキーのAグループで行う

図 1.44b　入力モードのチェンジ操作によるモード遷移図

長　音	
ローマ字	RO－MAZI*
ユーザー	YU－ZA－
つまる音	
がっこう(学校)	GAKKOU
いっち(一致)	ITTI
あっ	A SHIFT TU
うっ	U SHIFT TU

*－は「マイナス」または「ハイフン」記号

を、ん	
を	WO
ん	NN または N′または X
しんい(真意)	SINNI または SIN′I
ほんしゃ(本社)	HONSYA または HONNSYA または HON′SYA
かな小文字	
ぁ	SHIFT A
ぃ	SHIFT I
ゅ	SHIFT YU

あ	あ A	い I	う U	え E	お O	ぁ 注 2	ぃ 注 2	ぅ 注 2	ぇ 注 2	ぉ 注 2
か	か KA CA	き KI	く KU CU QU	け KE	こ KO CO	きゃ KYA	きぃ KYI	きゅ KYU	きぇ KYE	きょ KYO
さ	さ SA	し SI SHI	す SU	せ SE	そ SO	しゃ SYA SHA	しぃ SYI	しゅ SYU SHU	しぇ SYE SHE	しょ SYO SHO
た	た TA	ち TI CHI	つ TU TSU っ 注 1	て TE	と TO	ちゃ TYA CYA CHA	ちぃ TYI CYI	ちゅ TYU CYU CHU	ちぇ TYE CYE CHE	ちょ TYO CYO CHO
な	な NA	に NI	ぬ NU	ね NE	の NO	にゃ NYA	にぃ NYI	にゅ NYU	にぇ NYE	にょ NYO
は	は HA	ひ HI	ふ HU FU	へ HE	ほ HO	ひゃ HYA ふぁ FA	ひぃ HYI ふぃ FI	ひゅ HYU ふ FU	ひぇ HYE ふぇ FE	ひょ HYO ふぉ FO
ま	ま MA	み MI	む MU	め ME	も MO	みゃ MYA	みぃ MYI	みゅ MYU	みぇ MYE	みょ MYO
や	や YA		ゆ YU		よ YO	ゃ 注 3	ぃ 注 3	ゅ 注 3	ぇ 注 3	ょ 注 3
ら	ら RA LA	り RI LI	る RU LU	れ RE LE	ろ RO LO	りゃ RYA LYA	りぃ RYI LYI	りゅ RYU LYU	りぇ RYE LYE	りょ RYO LYO
わ	わ WA	ゐ WI	う WU	ゑ WE	を WO					
ん	ん NN N′ X									
が	が GA	ぎ GI	ぐ GU	げ GE	ご GO	ぎゃ GYA	ぎぃ GYI	ぎゅ GYU	ぎぇ GYE	ぎょ GYO
ざ	ざ ZA	じ ZI JI	ず ZU	ぜ ZE	ぞ ZO	じゃ ZYA JA JYA	じぃ ZYI JYI	じゅ ZYU JU JYU	じぇ ZYE JE JYE	じょ ZYO JO JYO
だ	だ DA	ぢ DI	づ DU	で DE	ど DO	ぢゃ DYA	ぢぃ DYI	ぢゅ DYU	ぢぇ DYE	ぢょ DYO
ば	ば BA	び BI	ぶ BU	べ BE	ぼ BO	びゃ BYA	びぃ BYI	びゅ BYU	びぇ BYE	びょ BYO
ぱ	ぱ PA	ぴ PI	ぷ PU	ぺ PE	ぽ PO	ぴゃ PYA	ぴぃ PYI	ぴゅ PYU	ぴぇ PYE	ぴょ PYO
ヴ	ヴぁ VA	ヴぃ VI	ヴ VU	ヴぇ VE	ヴぉ VO					

（この表は主要なものだけで、その他の綴り方は省略します）

〈注 1〉 SHIFT +TU、TSU で表示する　〈注 2〉 SHIFT +A、I、U、E、O で表示する　〈注 3〉 SHIFT +YA、YI、YU、YE、YO で表示する

表 1.3　松茸 V2 におけるローマ字入力の綴り方

さきほど、「応用 MS-DOS」という文字列を入力してみましたが、その時の「OUYOU」というローマ字による入力モードをもとに、ひらがな、カタカナ、半角カタカナ、登録語変換などの各種の変換と、その基本操作について解説しましょう。これらの動作や操作は、カタカナキーによる入力モードの場合においても同じです。

入力した「読み」(ひらがな)を、カタカナや、ひらがな、半角などに変換する作業——つまり、辞書ファイルを参照しない(登録語に変換しない)変換作業は、ファンクションキーの f･6 、 f･7 で行います。一方、漢字変換など、辞書ファイルを参照して登録語を取り出すものは XFER で行います。

では、 CTRL ＋ XFER をキー入力して、日本語入力モードにはいった画面から始めましょう。

A>■ ……スクリーンカーソル上で変換作業を行う(直接入力)

Rか　A↓　貼込　8:19　記号　複写　松茸ｶﾅ変換　半角変　ﾛｰﾏ字　半角　カナ　Rか

おうよう ……OUYOUと読みを入力
オウヨウ ……f･6 を入力(カタカナ変換)
ｵｳﾖｳ ……f･7 を入力(半角変換)
おうよう ……登録語変換は半角カタカナ、全角カタカナの状態では行えない f･7 、f･6 を入力してもとにもどす
応用 ……XFER で登録語に変換
負うよう ……XFER で次の登録語の候補に変換
選択：　1:応用　2:負うよう　3:追うよう　4:おうよう ……↓を押すと、画面最下行に候補の一覧が表示される。↓、↑キーもしくは数字キーで候補を選択する
おうよう ……BS キーを押すと読みを入力した状態にもどる
応用 ……再度、XFER キーで変換する

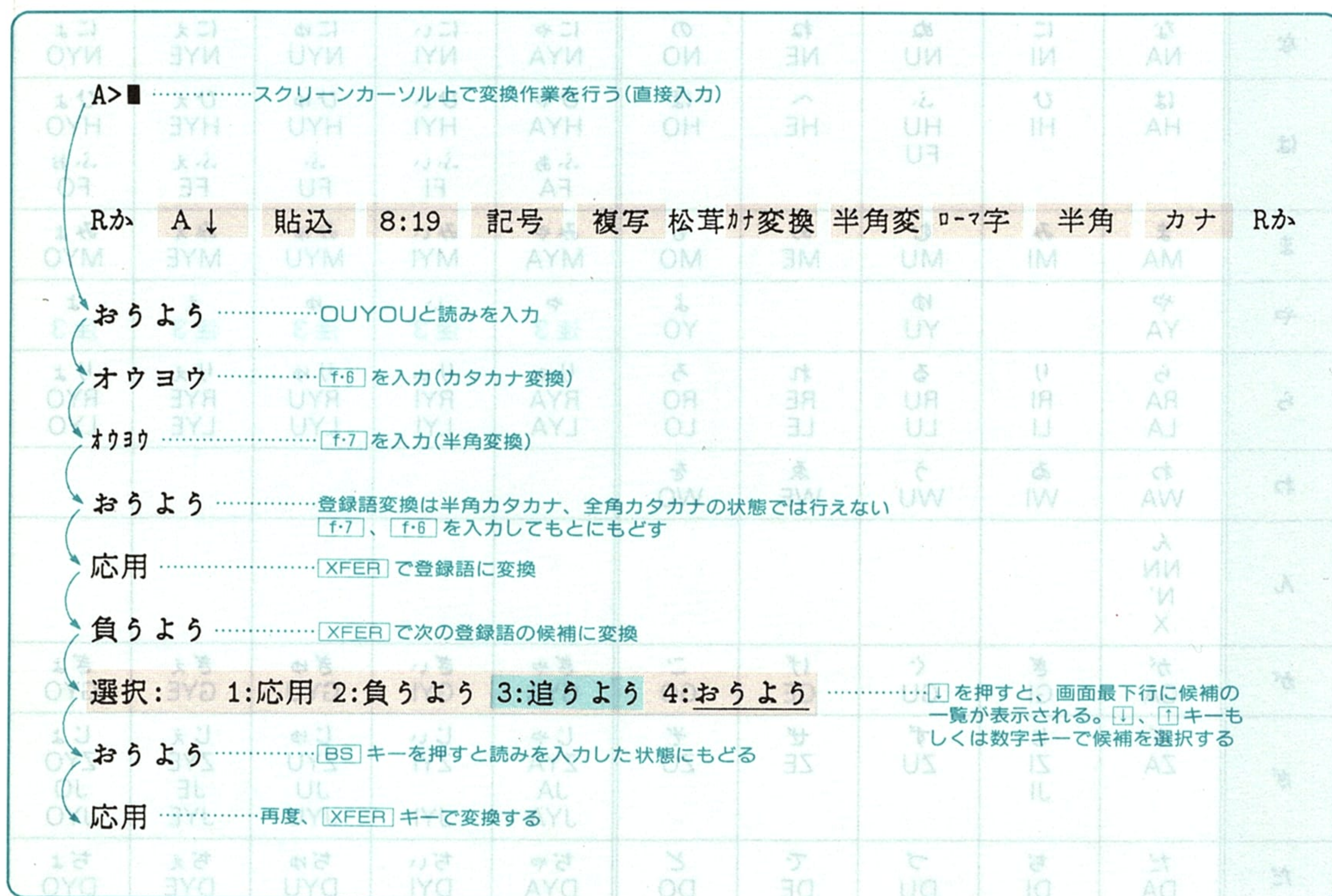



図 1.45　各種変換機能の基本操作

上の図には、入力した「読み」が、それぞれのファンクションキーや、 XFER によってさまざまに変換される様子が示されています。

BS 、DEL キーによる入力文字の削除

いずれの入力モードにおいても、入力された文字列は、BS キーによりカーソルの左側の文字を1文字ずつ削除、また DEL キーによりカーソル上の文字を1文字ずつ削除することができます。これらは、操作も動作も簡単ですので実行例は省略します。

カタカナの入力

カタカナを入力するには、2つの方法があります、1つは、普通の入力モードで「ひらがな」を入力して、それを A f･6 の「カナ変換」で「ひらがな→全角カタカナ」変換を行う方法。なお、これによって全角カタカナに変換した後、A f･7 により、半角カタカナに変換することができます。

もう1つの方法は、入力モードを A f･10 で「カタカナ入力モード」に切り替えて(メニューAの左右端の表示は「カ」となる)、ひらがな入力と同様に、ローマ字あるいはカタカナキーで、カタカナを直接入力できます。半角カタカナは、入力した全角カタカナを、A f･7 を押すことにより、半角に変換することができます。

また、半角のカタカナを直接入力するには、A f･9 で「半角英数字カナ入力モード」に切り替えて(メニューAの左右端の表示は「半」となる)、カナ キーを ON にしてカタカナキーで入力すれば可能です。では、それらの実行例を示しましょう。

ローマ字ひらがな入力モード

Rか　A↓　貼込　8:19　記号　複写　松茸ｶﾅ変換　半角変　ﾛｰﾏ字　半角　カナ　Rか

おうよう ……………OUYOUと読みを入力

オウヨウ ……………f･6 (カナ変換)を入力して、「ひらがな→全角カタカナ」変換を行う

ｵｳﾖｳ ……………………f･7 (半角変)を入力して、「全角→半角」変換を行う

ローマ字カタカナ入力モード(f･10 を入力)

Rカ　A↓　貼込　8:19　記号　複写　松茸ｶﾅ変換　半角変　ﾛｰﾏ字　半角　カナ　Rカ

オウヨウ ……………OUYOUと読みを入力。全角のカタカナに変換される

ｵｳﾖｳ ……………………f･7 (半角変)を入力して、「全角→半角」変換を行う

半角英数字カナ入力モード(f･9 を入力)

半　A↓　貼込　8:19　記号　複写　松茸ｶﾅ変換　半角変　ﾛｰﾏ字　半角　カナ　半

ｵｳﾖｳ ……………カナ キーをONにして、オウヨウと入力。直接半角カナが入力できる

図 1.46　カタカナの入力

英数字の入力

英数字は、A［f･8］でローマ字入力モードをOFFにすれば入力することができます（メニューAの左右端の表示の「Rか」、「Rカ」などの、ローマ字入力モードを示す「R」が表示されない状態）。もちろん［カナ］キーはOFFです。半角英数字は、入力した全角英数字を、A［f･7］を押すことにより、半角に変換することができます。

また、半角の英数字を直接入力するには、A［f･9］の「半角英数字カナ入力モード」で可能です（メニューAの左右端の表示は「半」となる）。このモードで入力した半角英数字は、A［f･7］で大文字に変換することができるほか、図1.45で示した操作により、漢字、全角ひらがな、全角／半角カタカナへも変換することができます。

半角文字（英数字、カナ、各種の記号など）を入力するには、松茸を使わずに（つまりFEPをOFFにして）通常入力で行えばよいのですが、松茸上で半角で入力された文字は、松茸を使わずに通常入力したものと同等に、MS-DOSやアプリケーションプログラムの各種のコマンドの入力に使うことができます。

その一例として、「半角英数字入力」モードで、MS-DOSのDIRコマンドを実行した場合を示しましょう。

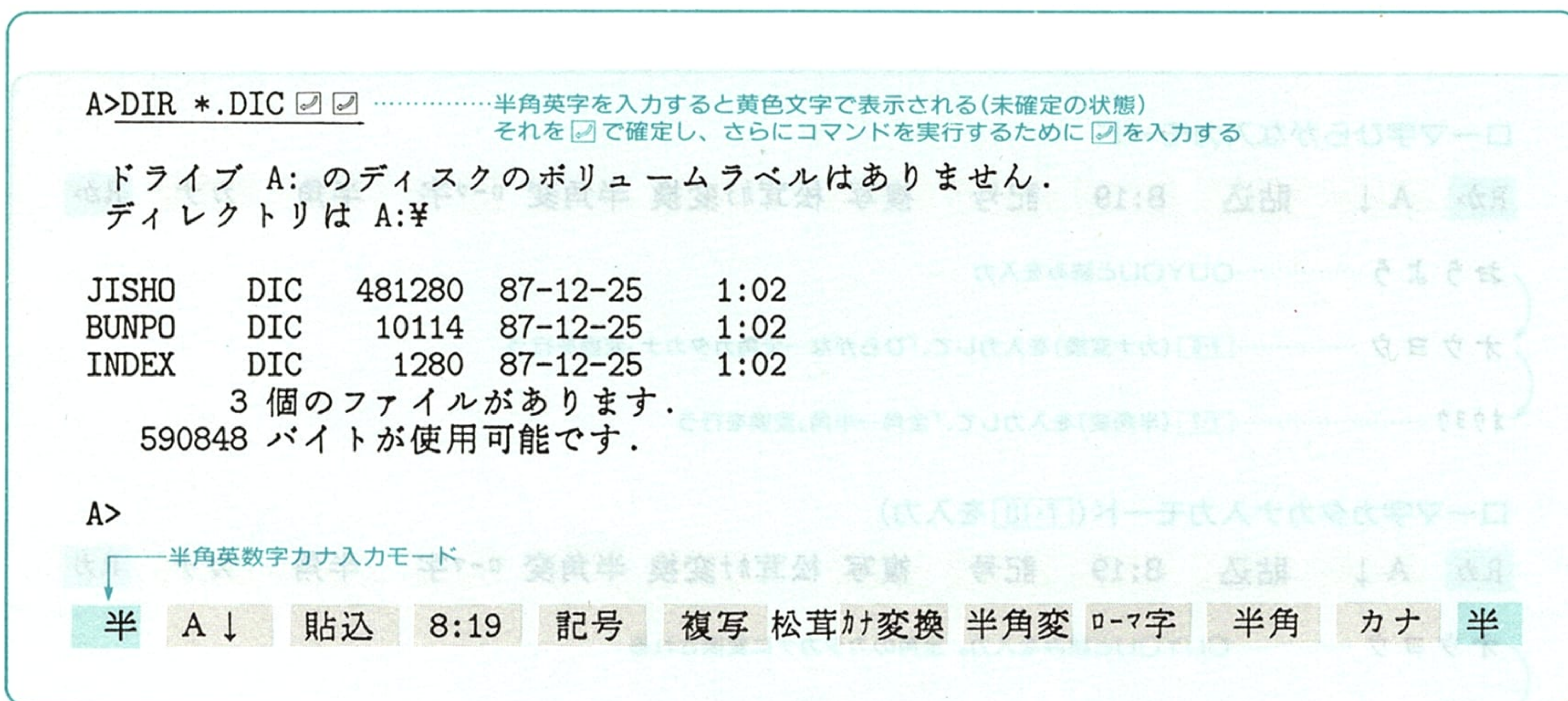
```
A>DIR *.DIC↵↵

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

JISHO    DIC   481280  87-12-25    1:02
BUNPO    DIC    10114  87-12-25    1:02
INDEX    DIC     1280  87-12-25    1:02
         3 個のファイルがあります.
   590848 バイトが使用可能です.

A>
```


図1.47　「半角英数字カナ入力」モードでMS-DOSコマンドを実行する

■ 連文節変換の入力例

松茸の連文節変換により、ワープロ感覚で文章を入力する一例を示しましょう。次の文をスクリーンカーソルの位置(ここでは MS-DOS のプロンプト、「A>」の後)に入力します。

1969 年 7 月 21 日、宇宙船アポロ 11 号のアームストロング飛行士らは、
月への着陸に成功し、その表面に初めて人類の足跡を残した。

ここでの入力は、「ローマ字入力モード」で行っています。文章を適切な連文節単位に区切りながら変換していくわけですが、この区切り方は、変換効率の面から重要なポイントになります。このノウハウは、それぞれの日本語入力システムによって異なり、これを簡単に解説することは困難です。使い込んだ経験から身につけるしかないでしょう。ただし、現在のどの日本語入力システム(ワープロを含めて)についても言えることは、あまりいくつもの文節を一度に変換しようとせず、せいぜい 2～3 の確実に変換できる範囲に区切って、こまめに変換することをお勧めします。その方が結局は能率的なのです。多くの文節からなる連文節の一部が正しく変換されなかった場合の修正は、非常にめんどうで時間がかかります。

連文節の入力に関して、松茸にはほかの FEP には見られない、たいへん便利な機能があります。それは「入力時に(つまり事前に)文節の区切りを松茸に指示する」機能です。その文節の区切りの指示は、スペースを挿入することにより行います。挿入した 1 個のスペースは、かな－漢字変換時に自動的に削除されますので、連続した連文節として変換されます。たとえば、

「きょう□はいる」→[変換]→「今日入る」
「きょうは□いる」→[変換]→「今日は居る」

というぐあいです。このように、FEP が文節の区切りを誤りそうな場合の「手助け」として、この機能を有効に利用することができます。

さて、次に示すのは、この文章をローマ字で入力する場合のキー入力の一例を、変換単位に区切って示したものです。ここでは修正作業を実習するために、故意に長い文節に区切っていますが、実際には、なるべくこまめに区切ることをお勧めします。

```
1969NEN7GATU21NITI、[XFER]
UTYUUSEN'APORO11GOUNO [XFER] (注1)
A−MUSUTORONGU [f·6]
HIKOUSIRAHA、[XFER] (注2)
TUKIHENOTYAKURIKUNI [XFER]
SEIKOUSI、[XFER]
SONOHYOUMENNI [XFER]
HAZIMETE [XFER]
ZINRUINOSOKUSEKIWONOKOSITA。[XFER] (注3)
```

図 1.48　連文節変換による実際のキー入力の一例

図 1.48 の変換において、注 1、注 2、注 3 の行の変換が一発では成功しませんでしたので、そのあとの修正操作の例を示します。

(注2)
ひこうしらは、
[XFER] ……………最初の文節を正しく変換
非行史らは、
[XFER]
飛行史らは、
[→] ……………2番目の文節へ移動
飛行史らは、
[XFER]
飛行氏らは、
[↓]
選択：　1:史　2:氏　3:士　4:死　5:子　6:師　7:紙　8:詩　9:誌 ……………選択モードへ入る
[3]
飛行士らは、
[↵]または次の語の読みを入力

(注3)
じんるいのそくせきをのこした。
[XFER]
人類の即席を残した。
[→] ……………2番目の文節へ移動
人類の即席を残した。
[XFER]
人類の足跡を残した。
[↵]または次の語の読みを入力

図 1.49a　連文節変換において、目的の語に変換されなかった場合の修正操作

このように黄色文字の部分は、再度 XFER を押すことにより再変換、または ↓ により変換候補群が表示されます。表示された候補群の選択は ↑ ↓ で行い ⏎ で確定します。さらに次候補群の表示、および前候補群へもどるには、SHIFT ＋ ↑、SHIFT ＋ ↓ で行います。

また、変換の対象となる文節を選択するには、← 、→ を入力することにより、松茸が区切った文節単位で、黄色文字の部分（変換の対象となる文節）が左右に移動します。さらに文節の区切り方を変更するには、BS を入力して、未確定の部分を最初の入力文字の状態にもどした後、← 、→ により、１文字単位で任意の区切り方に変更します。このとき、↑ で入力文字の左端、↓ で入力文字の右端にカーソルを移動することができます。

いずれの場合も、XFER による変換は、カーソルの左側の黄色文字が対象になり、カーソルの右側のものは黄色文字であっても対象になりません。

注1
うちゅうせんあぽろ１１ごうの ……………最初の入力
XFER
宇宙船亜ぽろ１１ごうの ……………このように変換された
→
宇宙船亜ぽろ１１ごうの ……………「アポロ」の文節が正しくない
BS
宇宙船あぽろ１１ごうの■ ……………一度「未確定」の状態にもどす
←
宇宙船あぽろ１１ごうの ……………「アポロ」の文節までカーソルを移動
f･6
宇宙船アポロ１１ごうの ……………カナに変換
→
宇宙船アポロ１１ごうの■ ……………カーソルを文末に移動
XFER
宇宙船アポロ１１号の ……………残りの文節を XFER で変換する

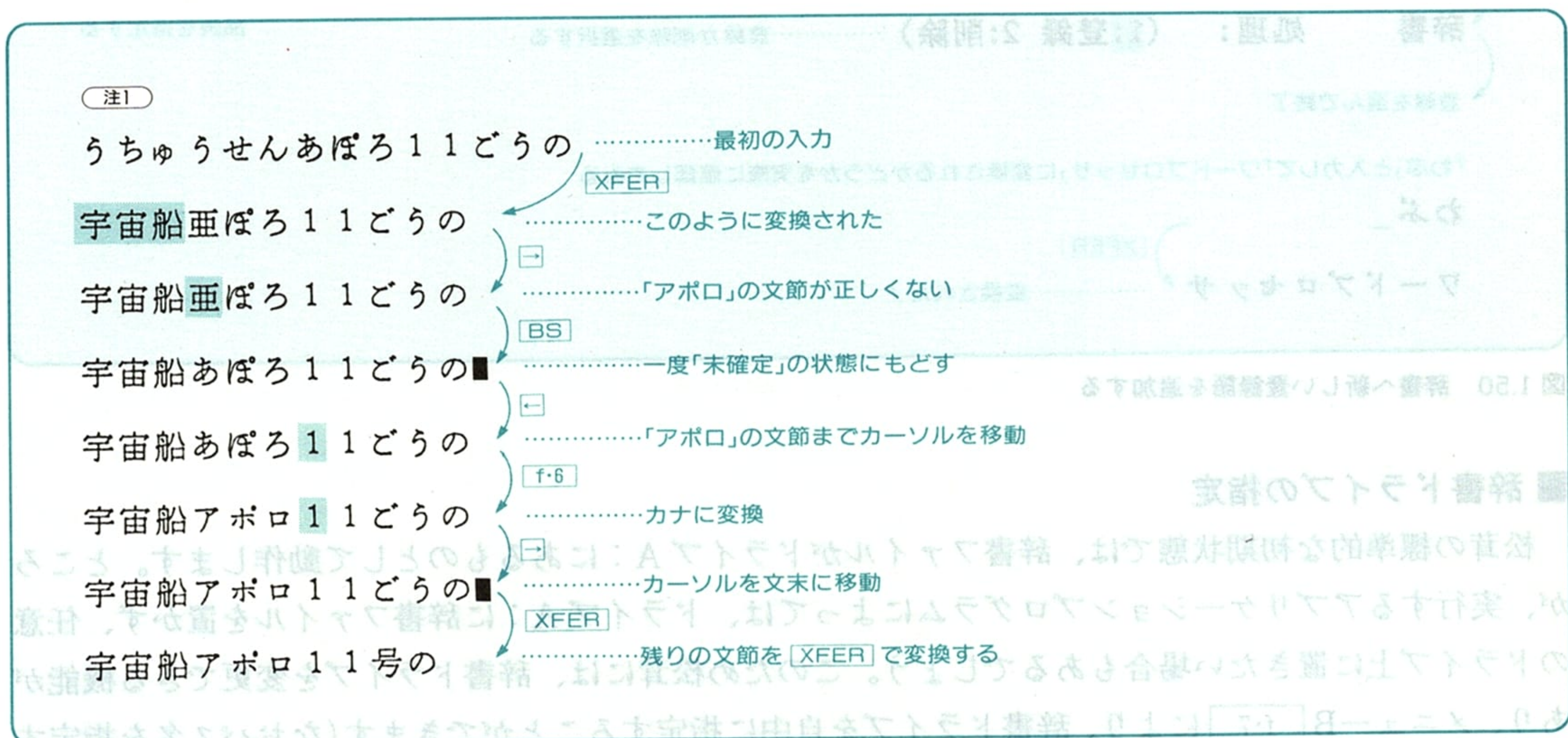



図 1.49b　松茸が文節の区切り方を誤った場合の訂正法

■ 辞書登録語の追加／変更機能

松茸の辞書の登録語の追加や変更は、メニューB f･5 によって行うことができます。では実行例として、「ワードプロセッサ」という単語を、「ワプ」という読みで登録してみましょう。

画面上の任意の位置にこの語が用意されているとする。メニューB f·5 を入力する

・・・・・ワードプロセッサ・・・・・ …単語の始めと終わりを指定する

辞書登録語の追加/変更

Rか　B↓　ﾌﾘｶﾞﾅ　ｺｰﾄﾞ表　JIS　辞書　先読　学習　ﾄﾞﾗｲﾌﾞ　9-05　電卓　モード　Rか

辞書　どこから？ …始点を指定して↵

辞書　どこまで？ …終点を指定して↵

カーソル位置の文字コードが表示される

文字コード JIS 256F $JIS 838F

文字コード JIS 2535 $JIS 8354

辞書　　読み：

辞書　　読み：わぷ↵ …「WAPU」と入力する

辞書　　品詞：　（1:一般名詞　2:人名　3:地名　4:固有名詞　5:サ変動詞　6:単漢字）…品詞を指定する

辞書　　処理：　（1:登録　2:削除）…登録か削除を選択する

登録を選んで終了

「わぷ」と入力して「ワードプロセッサ」に変換されるかどうかを実際に確認してみる

わぷ_

XFER

ワードプロセッサ …変換された！

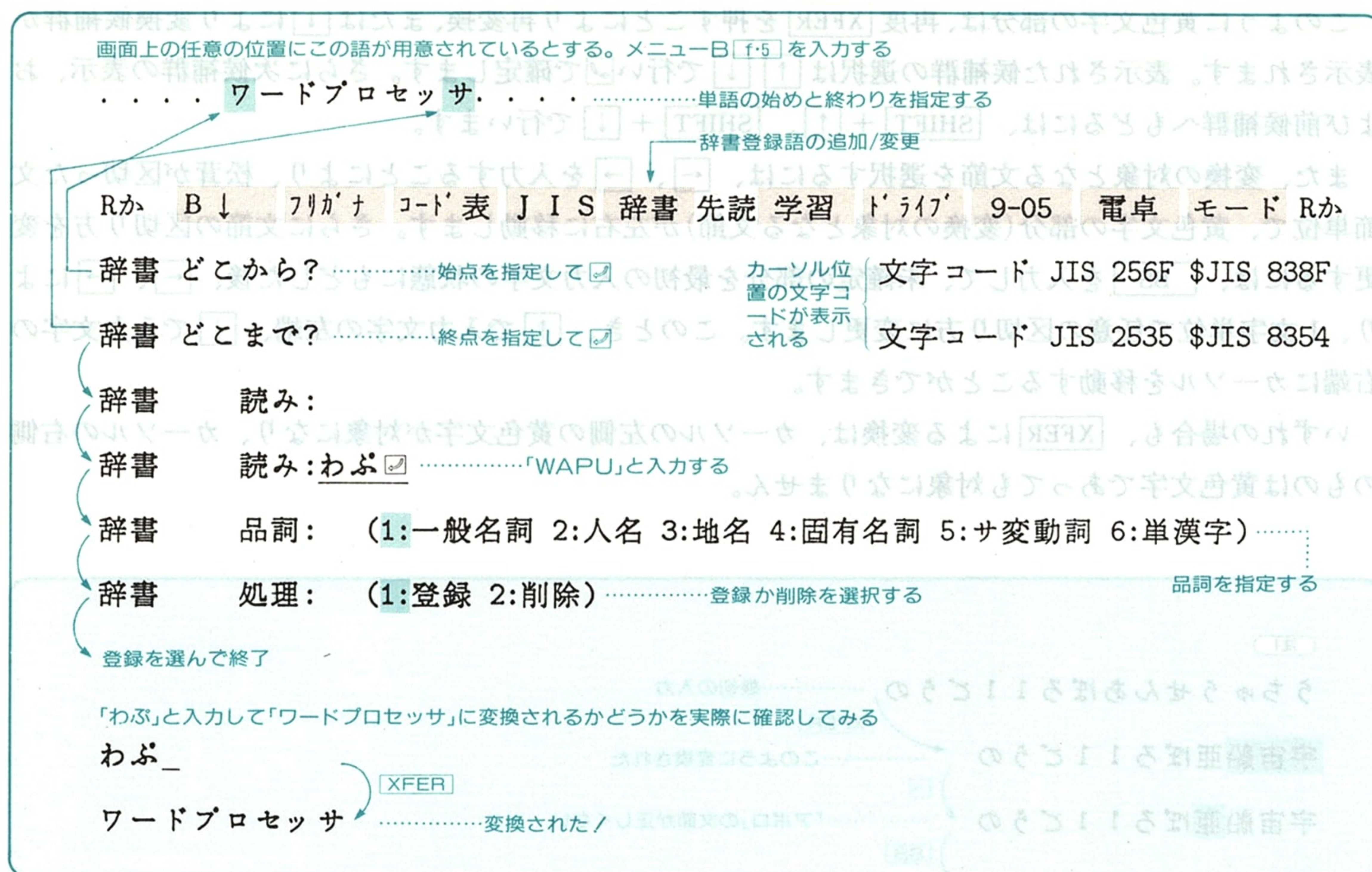



図 1.50　辞書へ新しい登録語を追加する

■辞書ドライブの指定

松茸の標準的な初期状態では、辞書ファイルがドライブA：にあるものとして動作します。ところが、実行するアプリケーションプログラムによっては、ドライブA：に辞書ファイルを置かず、任意のドライブ上に置きたい場合もあるでしょう。このため松茸には、辞書ドライブを変更できる機能があり、メニューB f·7 により、辞書ドライブを自由に指定することができます(なおパス名を指定することで，辞書ファイルを任意のディレクトリ上に置くことも可能)。では辞書ドライブを変更する実行例を示しましょう。

辞書ドライブの変更

Rか　B↓　ﾌﾘｶﾞﾅ　ｺｰﾄﾞ表　JIS　辞書　先読　学習　ﾄﾞﾗｲﾌﾞ　9-05　電卓　モード　Rか

f·7

辞書パス　B:¥ …任意のドライブ名を指定する。さらにパス名を変更することにより、辞書をどのドライブのどのディレクトリにも置くことができる。たとえば「E：¥DIC」などと指定できる

<f·2>:貼込

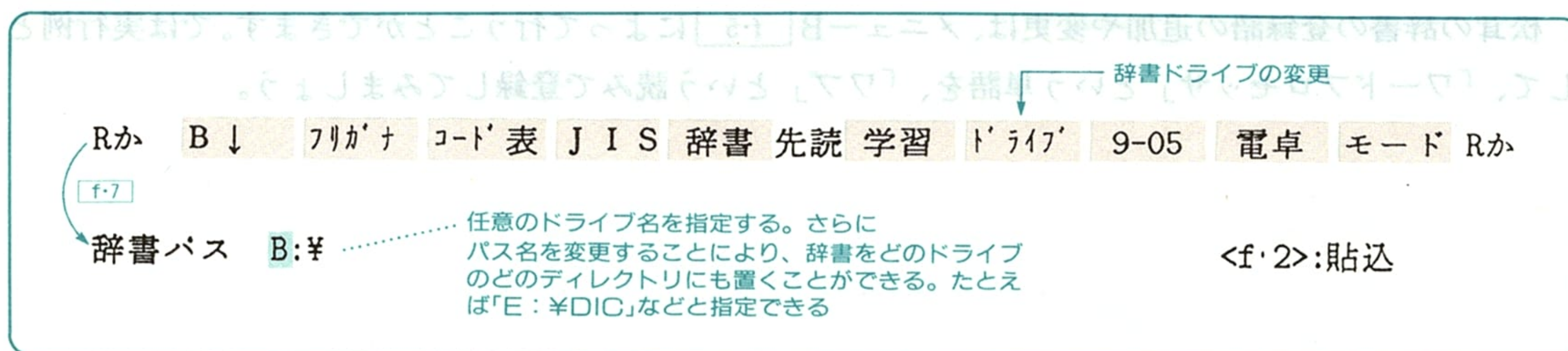



図 1.51　辞書ドライブの指定の変更

この操作以降は、ここで指定されたドライブ上の辞書ファイルが使われることになります。なお辞書パスの指定は、CONFIG.SYS ファイルへの登録によっても行うことができます。その場合は、松茸のデバイスドライバが組み込まれるときに設定されます。

■ 学習機能の ON／OFF

松茸では、かなー漢字変換や単字変換などの登録語への変換の際、同じ読みの登録語が複数存在する場合には、もっとも最近に使われた登録語の順に取り出されて変換されます。また、連文節の区切り方も、同じ語の前回の区切り方と同じように行われます。これは、自動学習機能(辞書ファイルの登録語の取り出し優先順位と、文節の区切り方を記憶しておく機能)が働いているからですが、この学習機能は、必要があれば OFF にすることができます。メニューB f･6 の「学習」を選択することにより、自動学習機能の ON／OFF を指定することができます。その実行例を示しましょう。

学習機能の設定モード
Rか　B↓　ﾌﾘｶﾞﾅ　ｺｰﾄﾞ表　J I S　辞書　先読　学習　ﾄﾞﾗｲﾌﾞ　9-05　電卓　モード　Rか
f･6
学習:　1.書込み　2.OFF　3.ON　[既定値　ON]　……選択は←→キーで行う
これは、現在までの学習結果を強制的に辞書に書き込む場合に使用する

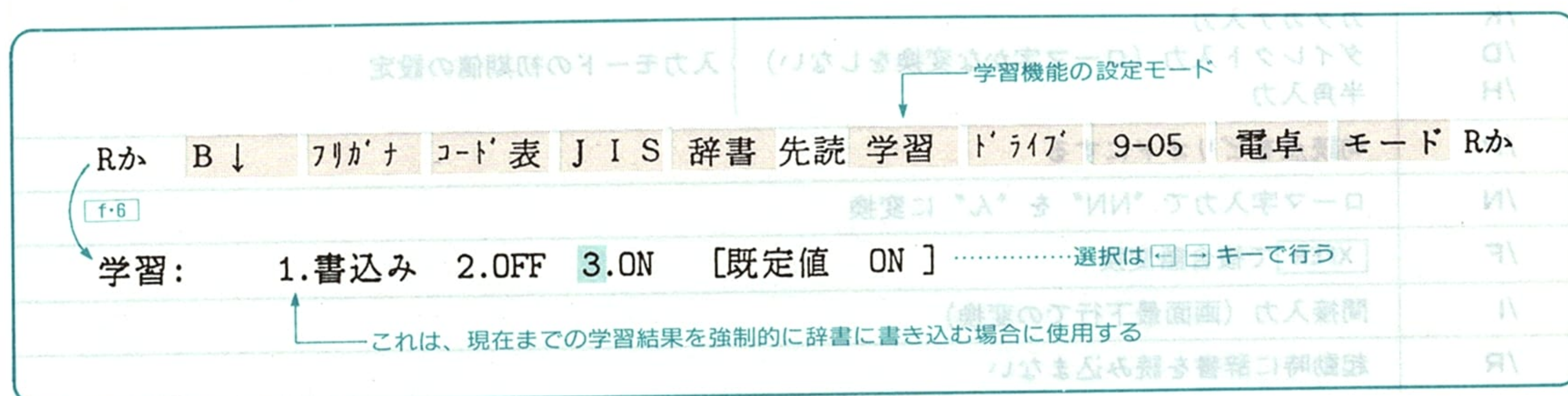



図 1.52　自動学習機能を ON／OFF する

■ CONFIG.SYS ファイルによる松茸の初期設定

CONFIG.SYS ファイルに登録する「DEVICE＝MTTK2.DRV」の行に、各種のパラメータを付加することにより、MS-DOS の起動時に、松茸の各種の動作やモードを、任意の状態に自動設定することができます。そのためのパラメータの一覧表を次ページに示します。

なお、これらのパラメータは、新松の補助ディスクに含まれている、松茸設定プログラム「MINSTALL.EXE」を実行することにより、簡単なメニュー形式で、目的の状態に設定することができます(CONFIG.SYS ファイルが自動的に書き換えられる)。

例：DEVICE＝MTTK2.DRV_B：¥DIC_/J5_/N_/#5_/L1

（「_」はスペースを表す）

- /L1 ― 文節の区切りを学習しない
- /#5 ― 5文字入力されるごとに辞書を引く
- /N ― ローマ字入力で〝NN〞を〝ん〞に変換
- /J5 ― 辞書のバッファサイズは50Kバイト
- B：¥DIC ― 辞書はドライブB：の[¥DIC]上のものを使う

パラメータ	機　　能
/1	実装されている漢字ROMが第1水準のみの場合に設定
/Jn	辞書のバッファサイズ。単位は10Kバイト（n≧1）
/K /D /H	カタカナ入力 ダイレクト入力（ローマ字かな変換をしない） 半角入力 （以上3つ）入力モードの初期値の設定
/P	句読点をピリオドにする
/N	ローマ字入力で〝NN〞を〝ん〞に変換
/F	XFER で複合語変換
/I	間接入力（画面最下行での変換）
/R	起動時に辞書を読み込まない
/T	松茸が起動していないときにも時計を表示
/U	半角入力モードで、1文字入力するごとに確定
/X$	変換モードの指定。$は1文字以上で、次の半角カナ文字を指定 ク：句読点（、。，．）の入力で、かな漢字変換を行う ハ：全角のひらがなを半角のカタカナに半角変換する ス：全角入力モード時に、スペースでかな漢字変換を行う ア：辞書の先読みを行わない カ：カーソル位置によらず、すべての未確定文字列を変換対象とする
/#n	辞書の先読みモード時、n文字（1≦n≦9）ごとに辞書を引く
/Ln	学習機能のモードを設定する。nは次の数値を指定 0：辞書および文節の区切りを学習しない 1：文節の区切りのみ学習しない 2：辞書および文節の区切りを学習する（標準初期状態）
/G	基本6品詞以外の、動詞、形容詞、連体詞、副詞などの品詞（システム辞書に関するもの）の登録／削除を可能にする。

表1.4　松茸の初期設定用の各種パラメーター覧表

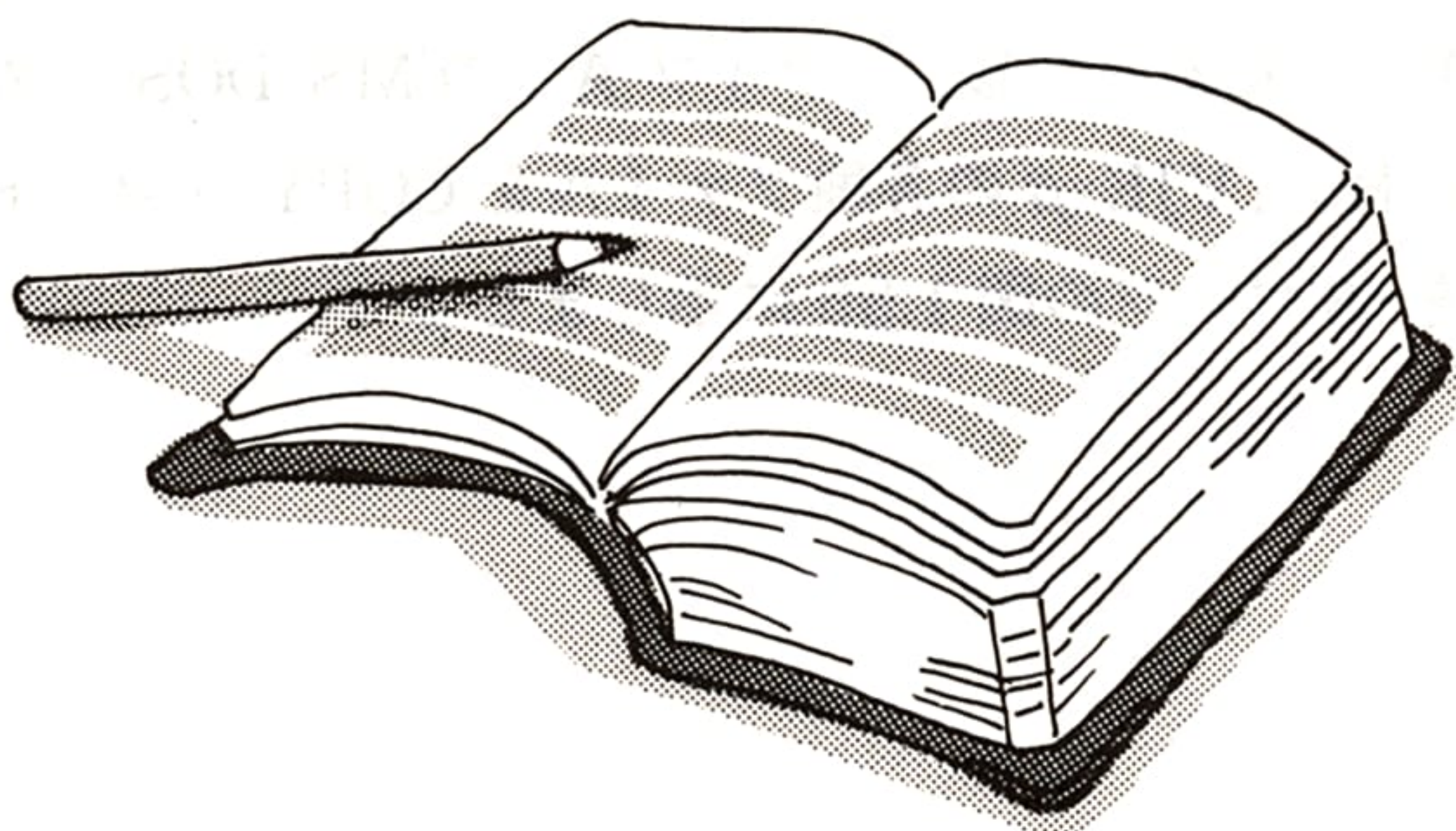
1.4 単独の日本語入力システム「VJE」（ブィジェイイー）

PC-9800シリーズ用のVJE(バックス社)を紹介しましょう。VJEは、ワープロソフトに付属している FEP ではなく、FEP として独立した製品です。VJE の歴史は、1984年に発売された「VJE-86」に始まります。VJEという名前の由来は、「Vacs Japanese Entry system」からとられています。また同じ年にバージョンアップされた「VJE-II」には、「日本語入力フロント・プロセッサ」という製品名がつけられ、これが日本語入力システムを表す言葉として現在広く使われている「FEP」(フロントエンドプロセッサ)の元祖となりました。

VJEのバージョンは、その後「VJE-α」、「VJE-Σ」と発展し、「VJE-β」で1つの安定期に達しました。ここではVJE-βをもとに解説しますが、それぞれのバージョンにおいても、操作の基本はほぼ共通です。VJE-βは、単独のFEPとして市販されていますが、前節のATOKや松茸と同様に、ユーザーの任意のディスクに組み込むことにより、ほかの多くのアプリケーションプログラムから、VJEの日本語入力機能を利用することができます。

■ VJE-βを組み込んだシステムディスクの作成

日常使用するMS-DOSのシステムディスクや、各種のアプリケーションプログラム上でVJE-βのFEPを利用できるように、VJE-βのオリジナルディスクに含まれる各種のファイルのなかから、必要なものをコピーする実例を示しましょう。VJE-βのバージョン2.0が動作するためには、VJE-βのみで約110Kバイトのメインメモリを占有しますので、考慮しておいてください。

さて、購入したVJE-βのオリジナルディスクには、次ページの図1.53のようなファイルが含まれています(MS-DOSのシステムは含まれていない)。

```
A>DIR B:
```
……VJE-βのオリジナルディスクに含まれるファイルの内容を見る

```
 ドライブ B: のディスクのボリュームラベルは VJE-β
 ディレクトリは B:¥
```

ファイル名	拡張子	サイズ	日付	時刻	説明
VJEB	SYS	76913	89-01-31	2:00	VJE-β本体のプログラム
VJEB	DRV	2763	88-10-10	2:00	VJE-βを起動するためのプログラム
VJEB	DIC	488448	89-01-08	2:00	辞書ファイル
VEDIT	EXE	34830	88-11-07	2:00	ユーザー辞書の登録/削除
VDISP	EXE	23503	88-09-19	2:00	辞書内容表示
VREFM	EXE	26777	88-09-19	2:00	辞書の再編成(整理)
VMERG	EXE	27473	88-09-19	2:00	2つの辞書の結合
VDIFF	EXE	27347	88-09-19	2:00	2つの辞書に共通でない登録語のみの辞書の作成
VCONV	EXE	20054	86-08-11	1:02	α用辞書をβ用に変換する
VFONT	EXE	49010	86-08-11	1:00	外字の作成
VLOAD2	COM	8440	88-11-07	2:53	外字をプリンタ内にロードする
SETVJE	EXE	15394	88-11-01	2:00	VJE-βの初期状態をCONFIG.SYSファイルに登録する
VCALCB	SYS	6495	88-09-19	2:00	電卓プログラム
VFILEB	SYS	6821	88-09-19	2:00	画面表示文字のセーブ/ロードプログラム
BVJEB	DIC	247808	89-01-08	2:00	2DDタイプのドライブ用の縮小版辞書
CONFIG	SYS	84	88-09-19	2:00	VJE-βのためのCONFIG.SYSファイル
SETUP	BAT	1525	88-09-19	2:00	VJE-βのシステムのコピー用バッチファイル
VJEWIN	EXE	12144	89-01-08	2:00	MS-WINDOWSでVJE-βを使用するための実行ファイル

（VEDIT〜SETVJE：それぞれのプログラムを直接実行するユーティリティプログラム）

（VCALCB・VFILEB：CONFIG.SYSファイルに登録して実行する。直接実行はできない。）

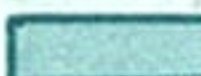
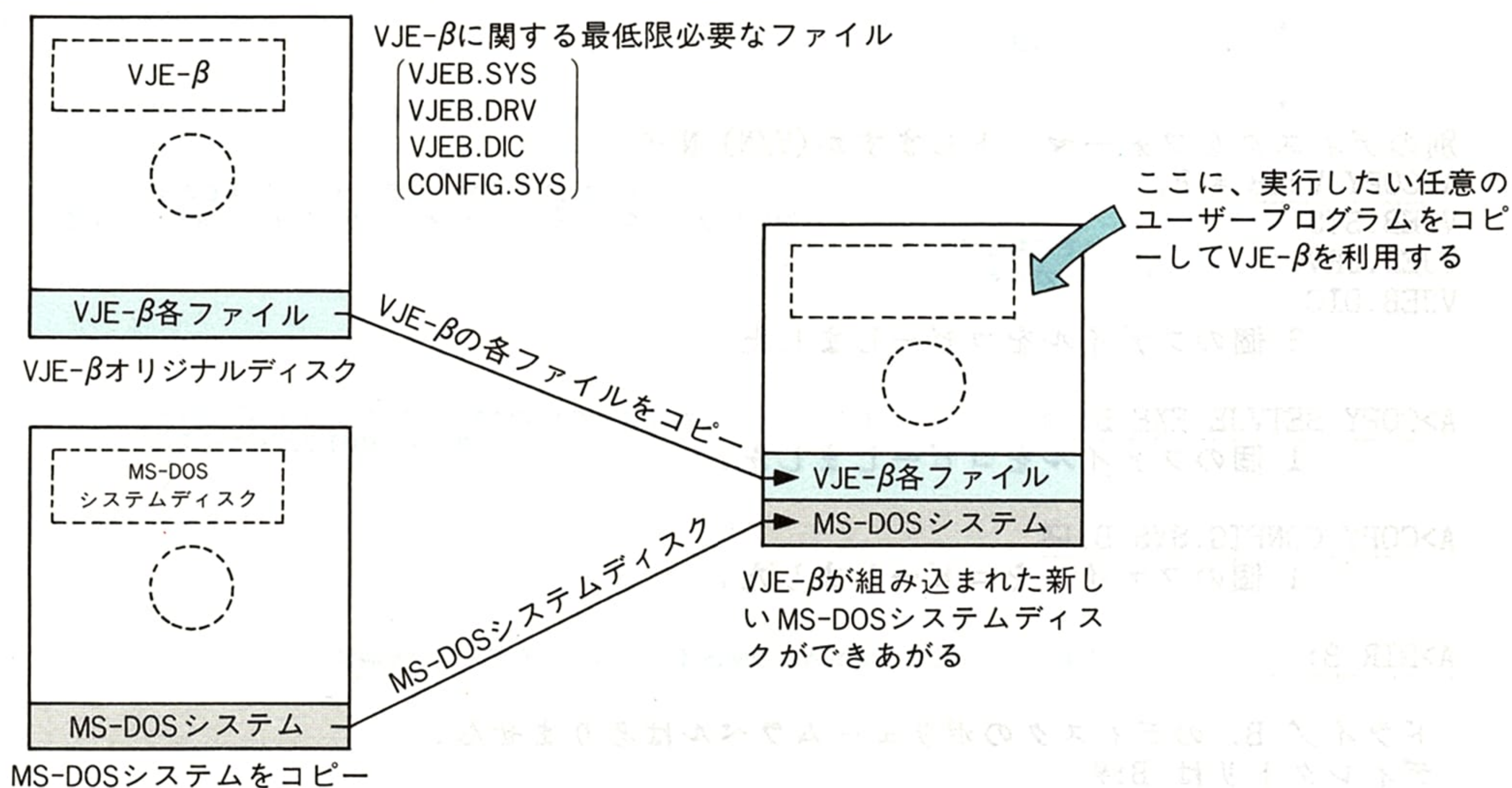
```
       18 個のファイルがあります.
    164864 バイトが使用可能です.

A>
```

■のファイルがVJE-βが動作するために必要な最少限のファイル

図 1.53　VJE-β のオリジナルディスクに含まれる各種ファイル

まず、フォーマットした空ディスクに MS-DOS システム(COMMAND.COM を含む)をコピーし、その上に VJE-β の動作に最低限必要な各種ファイル(図の色付きのファイル)をコピーして、VJE-β を組み込んだ新しい MS-DOS システムディスクを作成してみましょう。

ドライブ A：に日常使っている MS-DOS システムディスク、ドライブ B：にコピー先のディスクをセットします。そして「/S」付きの FORMAT コマンドを実行し、MS-DOS システムをコピーします。この作業が終わった後、ドライブ A：の MS-DOS システムディスクを VJE-β のオリジナルディスクと入れ替え、VJE-β の各種ファイルを COPY コマンドでドライブ B：にコピーします。この一連の実行例を 68 ページの図 1.55 に示しましょう。

〈VJE-βの組み込み〉

■ 例1. 新しいMS-DOSシステムディスクを作成する場合

■ 例2. すでに使用中のユーザーディスク上にコピーする場合

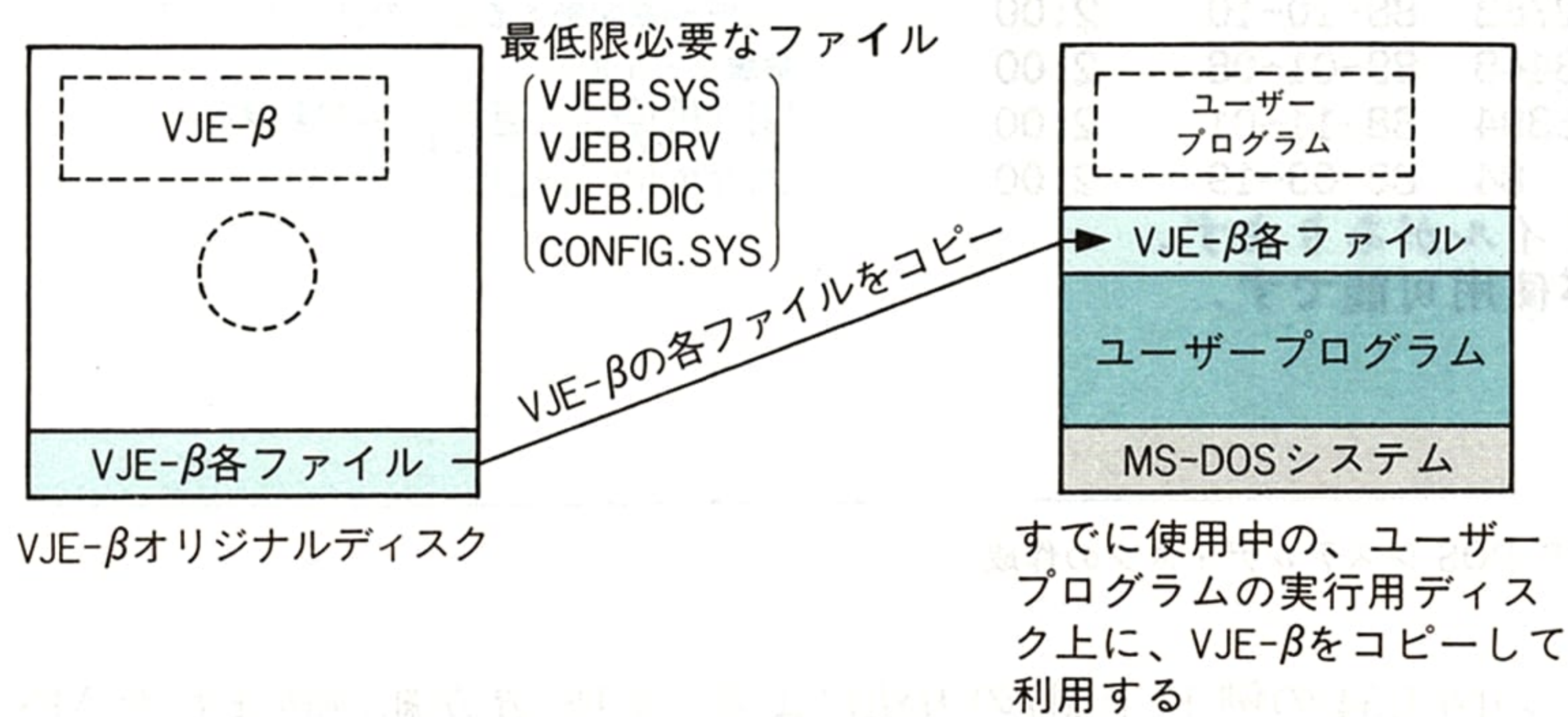



図 1.54　VJE-β を移植する 2 つの方法

```
(ドライブA:にMS-DOSシステムディスクをセットする)

A>FORMAT B:/S⏎ ……………ドライブB:上のディスクをフォーマットし、ドライブA:上の
                          ディスクのMS-DOSシステムをコピーする
         •
         •  (途中のメッセージ省略)
         •
         •
別のディスクをフォーマットしますか(Y/N) N⏎
A>COPY VJEB.* B:⏎            (ドライブB:上にMS-DOSシステムディスクができあがった。
VJEB.SYS                      次にドライブA:にVJE-βのオリジナルディスクをセットする)
VJEB.DRV  } ワイルドカードでこれらの
VJEB.DIC    ファイルがコピーされる
        3 個のファイルをコピーしました.

A>COPY SETVJE.EXE B:⏎ ……………これはコピーしておかなくてもよいが、あればCONFIG.SYS
        1 個のファイルをコピーしました.           ファイルの設定が楽に行える

A>COPY CONFIG.SYS B:⏎
        1 個のファイルをコピーしました.

A>DIR B:⏎ ……………できあがったVJE-β組み込みのMS-DOSシステムディスクを確認する

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

COMMAND  COM     24931  88-07-13    0:00
VJEB     SYS     76913  89-01-31    2:00 ……………VJE-β本体のプログラム
VJEB     DRV      2763  88-10-10    2:00 ……………VJE-βを起動するためのプログラム
VJEB     DIC    488448  89-01-08    2:00 ……………辞書ファイル
SETVJE   EXE     15394  88-11-01    2:00 ……………CONFIG.SYSファイル自動設定
                                                 プログラム(なくてもよい)
CONFIG   SYS        84  88-09-19    2:00 ……………CONFIG.SYSファイル
        6 個のファイルがあります.
   542720 バイトが使用可能です.

A>
```

図 1.55　VJE-β を組み込んだ MS-DOS システムディスクの作成

以上の作業で、前ページの図 1.54 の例 1(上側)の方法による、VJE-β が組み込まれた MS-DOS システムディスクができました。現在このディスクには、MS-DOS の本体と、VJE-β が動作するために必要な最低限のファイルのみが含まれています。実務には、このディスク上に各種のアプリケーションプログラムをコピーして使用するわけです。

またこのように、新しいシステムディスクを作成するのではなく、日常使用しているアプリケーションプログラムのシステムディスク上に、VJE-β の各種ファイルをコピーする、図 1.54 の例 2(下側)の方法もあります。その場合、ほかの FEP に関する各種のファイルが存在していれば、ディスクの容量を考慮して、事前にその FEP に関するファイルを削除しておきます(ハードディスクなどで、容量に

十分な余裕があればその必要はない)。

いずれの場合も、CONFIG.SYSファイルには、VJE-β用のデバイスドライバを登録するための記述が必要です。そのCONFIG.SYSファイルの内容を、**図1.56**にTYPEコマンドで示しておきます。

では、作成したVJE-βのシステムディスクをドライブA：にセットし、リセットボタンを押してMS-DOSを起動し直してみましょう。ほかのFEPと同様に、MS-DOSを再起動することにより、VJE-βが動作可能となります。**重要!**

```
(リセットボタンを押す)
NEC PC-9800 Series Personal Computer

ﾏｲｸﾛｿﾌﾄ MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30
Copyright 1981,88 Microsoft Corp. / NEC Corporation

日本語入力フロント・プロセッサ ＶＪＥ－β 第２．００版
Copyright (C) 1986-88 VACS Corp. / ASCII Corp.

Command ﾊﾞｰｼﾞｮﾝ 3.30
現在の日付は 1989-08-09 (水) です.
日付を入力してください:
現在の時刻は 20:04:04.00 です.
時刻を入力してください:
(MS-DOSが完全に起動した)
A>TYPE CONFIG.SYS
FILES=10
BUFFERS=20
DEVICE = VJEB.DRV /HR /M1 /T0 /L /J /K- /K, /K. /D4

A>
```


図1.56　VJE-βを組み込んだシステムディスクによるMS-DOSの起動とCONFIG.SYSの内容

MS-DOSのオープニングメッセージと、VJE-βがデバイスドライバとして組み込まれたことを示すメッセージが表示されています。この画面のなかで、CONFIG.SYSファイルの内容をタイプアウトしていますが、その内容の最後の行が、VJE-βをMS-DOSに組み込むために登録しなければならない、最低限必要な部分です。またこの行のパラメータの内容によって、VJE-βのさまざまな初期モードを任意に設定することができますが、この機能については後述します。CONFIG.SYSファイルに関しては、1.1章および3.1章を参照してください。

■ VJE-β による日本語入力の操作法

VJE-β による日本語入力の操作の基本を実習しましょう。VJE-β の場合も、ほかの FEP と同様に、一般的にはローマ字による入力を強くお勧めします。無理・苦労してカタカナキーの入力を覚える必要はなく、ローマ字入力ですべて事足ります。

VJE-β の日本語入力モードには、まず大きな選択肢として、

- **メニューラインモード（間接入力モード）**
- **スクリーンモード（直接入力モード）**

の2つがあります。

メニューラインモードというのは、画面の最下段に日本語入力ラインが設けられ、このライン上で、「読み」を入力したり、かなー漢字変換を行ったりして、目的とする文字や文字列を準備してから、スクリーンカーソルの位置へ移す入力形式です。ATOK の標準モードなどと同様の形式です。

スクリーンモードというのは、日本語入力ラインを設けずに、「読み」の入力も、かなー漢字変換も、スクリーンカーソルの位置で直接行う入力形式です。どちらの形式を使うかは、VJE-β を利用するアプリケーションプログラムの都合によって選択します。一般的にはスクリーンモードの方が使いやすいのですが、アプリケーションプログラムによっては、メニューラインモードでなければうまく入力できない場合があるかもしれません。ここでは、画面下部にファンクションメニューが表示されるなど、学習するためにも都合のよい、スクリーンモードを主体に解説しますが、メニューラインモードの場合も基本的な操作は同じです。

では実習を始めましょう。まず通常入力モード（VJE-β が機能しない状態）から、VJE-β が機能する日本語入力モードにはいるには、そのスクリーンモードへ移る場合は CTRL ＋ XFER、メニューラインモードへ移る場合は SHIFT ＋ XFER を入力します。また、スクリーンモードとメニューラインモードの間も、同じキーを入力することで直接移行します。この関係を次の図で示しましょう。

通常入力モード
CTRL ＋ XFER
SHIFT ＋ XFER
スクリーンモード（直接入力）
スクリーンカーソルの位置に直接入力され、そこで変換される
CTRL ＋ XFER
SHIFT ＋ XFER
メニューラインモード（間接入力）
画面下部の日本語入力ラインに入力され、そこで変換され、その後スクリーンカーソルの位置に移される

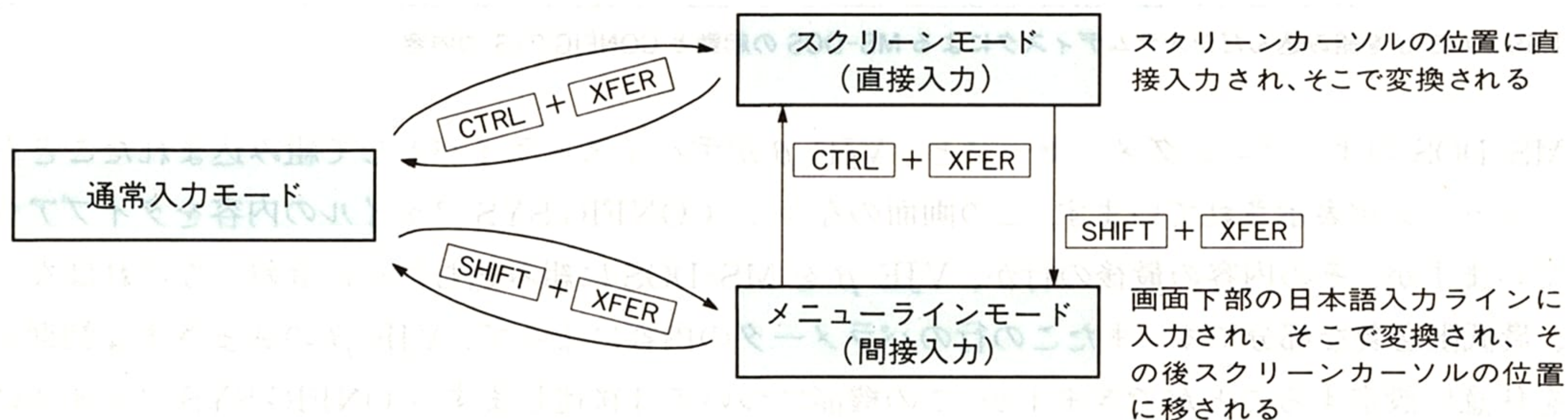



図 1.57　通常入力モード／スクリーンモード／メニューラインモードの切り替え

ここでは、とりあえずスクリーンモードによる日本語入力について話を進めます。まず CTRL ＋ XFER を入力し、VJE-β のスクリーンモードでの簡単な入力例を示し、また SHIFT ＋ XFER を入力し、VJE-β のメニューラインモードでの簡単な入力例を示しましょう。

CTRL ＋ XFER の入力によりVJE-βのスクリーンモードにはいった初期画面

A>■……………スクリーンカーソル

学習機能ON
JISコード
全角
入力時ひらがな変換
ローマ字入力

画面下部のメニューライン

かな　カナ　全角　半角　ｺｰﾄﾞ　10確定　^3ﾛｰﾏ字　^10その他　　Rかな全J*

f·1　f·2　f·3　f·4　f·5　f·10　CTRL ＋ f·3 ローマ字入力モードのON/OFF　CTRL ＋ f·10 その他の機能の選択

入力文字の変換モード

現在の各種モードの表示(ただし、初期モードはCONFIG.SYSの内容によって異なってくる)

図 1.58a　VJE-β のスクリーンモードにはいった初期画面

ローマ字→かな変換入力モードの場合のかな-漢字変換例

A>おうよう■……………OUYOUとキー入力

XFER

応用

……………MS-DOSとキー入力

応用ＭＳ－どＳ■

f·3

応用ＭＳ－ＤＯＳ■

f·10 または ⏎ により確定する

かな　カナ　全角　半角　ｺｰﾄﾞ　10確定　^3ﾛｰﾏ字　^10その他　　Rかな全J*

図 1.58b　スクリーンモードにおける入力例

SHIFT ＋ XFER の入力によりVJE-βのメニューラインモードにはいった初期画面

A>■

スクリーンカーソル

現在の各種モードの表示（図1.58a のものと同様）

入力 ■　日本語入力ライン（全角32文字まで入力可能）　　Rかな全J*

図 1.59a　VJE-β のメニューラインモードにはいった初期画面

ローマ字→かな変換入力モードの場合のかな-漢字変換例

A>応用ＭＳ－ＤＯＳ

入力 おうよう■ ……OUYOUとキー入力　Rかな全J*

XFER

入力 応用　Rかな全J*

……MS-DOSとキー入力

入力 ＭＳ－どS■　Rかな全J*

f･3

入力 ＭＳ－ＤＯＳ■　Rかな全J*

f･10 または

入力 ■　Rかな全J*

図 1.59b　メニューラインモードにおける入力例

入力モードの選択

VJE-β の入力モードには、

- 半角の英数を入力する**半角英数モード**
- 全角の英数を入力する**全角英数モード**
- 半角のカタカナをカタカナキーで入力する**半角カタカナ(カナキー)モード**
- 半角のカタカナをローマ字で入力する**半角カタカナ(ローマ字)モード**
- 全角のひらがなをローマ字で入力する**全角ひらがな(ローマ字)モード**
- 全角のひらがなをカタカナキーで入力する**全角ひらがな(カナキー)モード**
- 全角のカタカナをローマ字で入力する**全角カタカナ(ローマ字)モード**
- 全角のカタカナをカタカナキーで入力する**全角カタカナ(カナキー)モード**

の８種類のモードがあります。ただし、これらはあくまで、キー入力した際に最初に表示される字体を指定する入力モードであり、どのモードで(どの字体で)入力しても、そのあとで、ひらがな／カタカナ／登録語／全角／半角などに自由に変換できます。これらの入力モードの選択は、次に示すキー操作で行います。

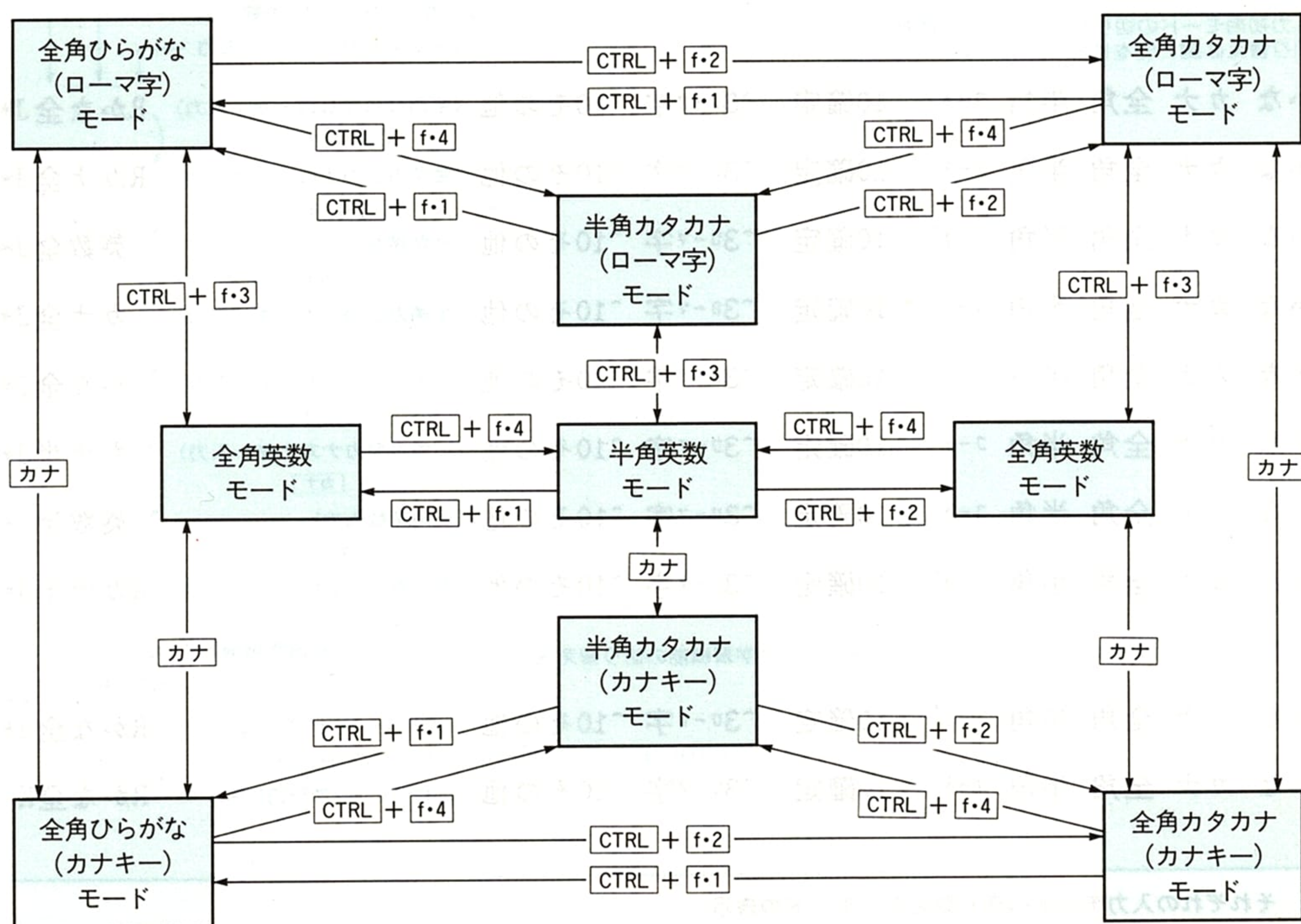



図 1.60　入力モードのチェンジ操作によるモード遷移図

現在の入力モードが何であるかの表示は、常に画面下部のメニューライン右端に、次ページの図 1.61 のように表示されています。各自でキー操作を行ってモードを切り替え、それぞれの状態を確認しておいてください。

これらの入力モードは、どのモードによる入力でも各種変換は自由に行えます。したがって、用途に合ったもっとも使いやすいモードにセットしておき、必要に応じてモードを切り替えればよいでしょう。なお、MS-DOS が起動したときに自動設定される初期モードは、CONFIG.SYS ファイルに登録する VJE-β のデバイスドライバのパラメータにより、任意のモードに設定することが可能です（後述）。

全角/半角
ひらがな/カタカナ/英数入力
ローマ字入力/カナキー入力

入力初期モードの切り替えの一例。その他への切り替えは図1.62を参照

かな カナ 全角 半角 ｺｰﾄﾞ 10確定 ^3ﾛｰﾏ字 ^10その他 (全角ひらがなローマ字入力) Rかな全J*
CTRL + f･2
かな カナ 全角 半角 ｺｰﾄﾞ 10確定 ^3ﾛｰﾏ字 ^10その他 (全角カタカナローマ字入力) Rカナ全J*
CTRL + f･3
かな カナ 全角 半角 ｺｰﾄﾞ 10確定 ^3ﾛｰﾏ字 ^10その他 (全角英数入力) 英数全J*
カナ
かな カナ 全角 半角 ｺｰﾄﾞ 10確定 ^3ﾛｰﾏ字 ^10その他 (全角カタカナカナキー入力) カナ全J*
CTRL + f･1
かな カナ 全角 半角 ｺｰﾄﾞ 10確定 ^3ﾛｰﾏ字 ^10その他 (全角ひらがなカナキー入力) かな全J*
CTRL + f･4
かな カナ 全角 半角 ｺｰﾄﾞ 10確定 ^3ﾛｰﾏ字 ^10その他 (半角カタカナカナキー入力) カナ半J*
カナ
かな カナ 全角 半角 ｺｰﾄﾞ 10確定 ^3ﾛｰﾏ字 ^10その他 (半角英数入力) 英数半J*
CTRL + f･3
かな カナ 全角 半角 ｺｰﾄﾞ 10確定 ^3ﾛｰﾏ字 ^10その他 (半角カタカナローマ字入力) Rカナ半J*

CTRL + f･10 の「その他」を選択した後、文字コードと学習機能の切り替えを行う

辞書学習機能 ON/OFF
区点/JIS

かな カナ 全角 半角 ｺｰﾄﾞ 10確定 ^3ﾛｰﾏ字 ^10その他 (JISコード/学習ON) Rかな全J*
CTRL + f･10 → f･4 → f･5 → CTRL + f･10
かな カナ 全角 半角 ｺｰﾄﾞ 10確定 ^3ﾛｰﾏ字 ^10その他 (区点コード/学習OFF) Rかな全K

図 1.61　それぞれの入力モードの切り替えと、モードの表示

かなー漢字変換モードの選択

VJE-β には、次の3つの変換モードがあります。

連文節……………XFER (かなー漢字変換キー) が入力された時点で変換作業が行われ、結果が表示される。

先読み……………入力される文章を自動的に解析しながら、内部的には変換作業の準備が行われており、XFER の入力により変換結果が表示される。

べた書き…………入力される文章を自動的に解析し、XFER の入力に関係なく、どんどん自動的に変換していく(自動変換)。

これらのモードの選択は、CTRL ＋ f･10 により「その他」のメニューを表示した後、f･6 で行います。その実行例を次に示します。なお、本書での実行例は、「連文節」モードで解説を行っています。

CTRL+f·10 により、このような「その他」のメニューが表示される。この後、f·6 により選択する

1:単語登録 2:辞書変更 3:キー設定 4:コード=JIS 5:学習=ON 6:変換=連文節

1:単語登録 2:辞書変更 3:キー設定 4:コード=JIS 5:学習=ON 6:変換=先読み

1:単語登録 2:辞書変更 3:キー設定 4:コード=JIS 5:学習=ON 6:変換=べた書

f·6

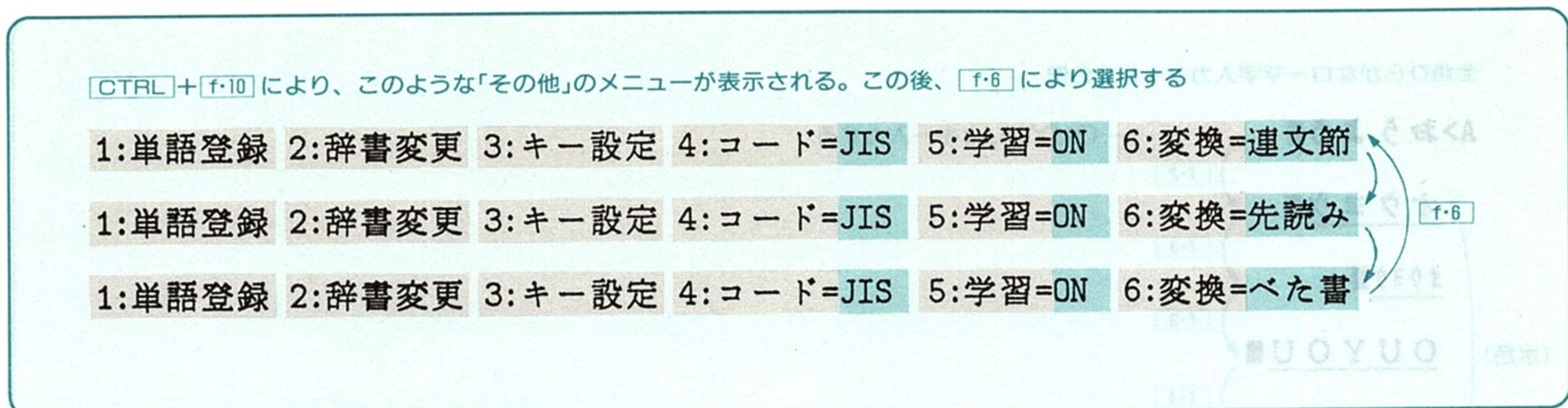



図 1.62　それぞれの変換モードにおけるモード表示

各種変換の基本操作

ではまず、入力モードを「全角ひらがなローマ字入力」に設定しておき、入力文字に対する各種変換操作の基本的な実行例を示しましょう。ここでは便宜上、スクリーンカーソル(MS-DOSのプロンプト「A>」)の位置に入力しますが、実際には各種のビジネスソフトやエディタなどのアプリケーションプログラムの画面で行うわけです。

まず、ローマ字で「OUYOU」とキー入力します。各種の変換操作は、f·1 ～ f·4 と、XFER、↓、↑、→、← で行います。ミスタイプの場合は、BS によりカーソルの左側の文字、DEL によりカーソル位置の文字を1文字ずつ削除することができます。また、入力した文字が黄色の状態の場合などは、ESC により入力文字全体を取り消すことができます(反転文字や、アンダーラインの状態の場合は、SHIFT ＋ ESC で取り消せる)。

このように、入力された文字がさまざまな形に変換されます。なおこれらは、入力モードに関係せず、それぞれの変換が可能です。また、ローマ字入力を使わずに、カナ キーをONにして、カタカナ入力によって、直接カタカナやひらがなを入力するモードでも同じです。

目的の文字列に変換した後、それを固定(確定)するには、

- f·10 の入力(CONFIG.SYSファイルに「/KR」パラメータを指定した場合はリターンキー)
- 次の「読み」の入力(何らかの変換操作を行った後、つまり入力文字が黄色でない場合に有効)
- 日本語入力モードの終了、またはモードの変更(CTRL ＋ XFER、または SHIFT ＋ XFER の入力)

のいずれかの操作により行います。固定されると、その部分が白色に変わり、他への変換はいっさいできなくなります。ただし NFER により、確定直後であれば再変換が可能です。

全角ひらがなローマ字入力モードでの例

A>おうよう■ ……………OUYOUとキー入力する

f・2

オウヨウ■

f・4

ｵｳﾖｳ■

f・3

（水色）　ＯＵＹＯＵ■

f・4

OUYOU■

f・1

おうよう■

ESC または ← → →

（黄）　おうよう■ ……………最初の状態と同じ

XFER

（水色）　応用

ESC

（黄）　おうよう■

スペースバー または → → ESC

（水色）　応用■

ESC

（黄）　おうよう■ ……………最初の状態と同じ

（　）内は文字の色
＿＿ はアンダーライン
▒▒ は反転文字

かな　カナ　全角　半角　ｺｰﾄﾞ　10確定　^3ﾛｰﾏ字　^10その他　　Rかな全J*

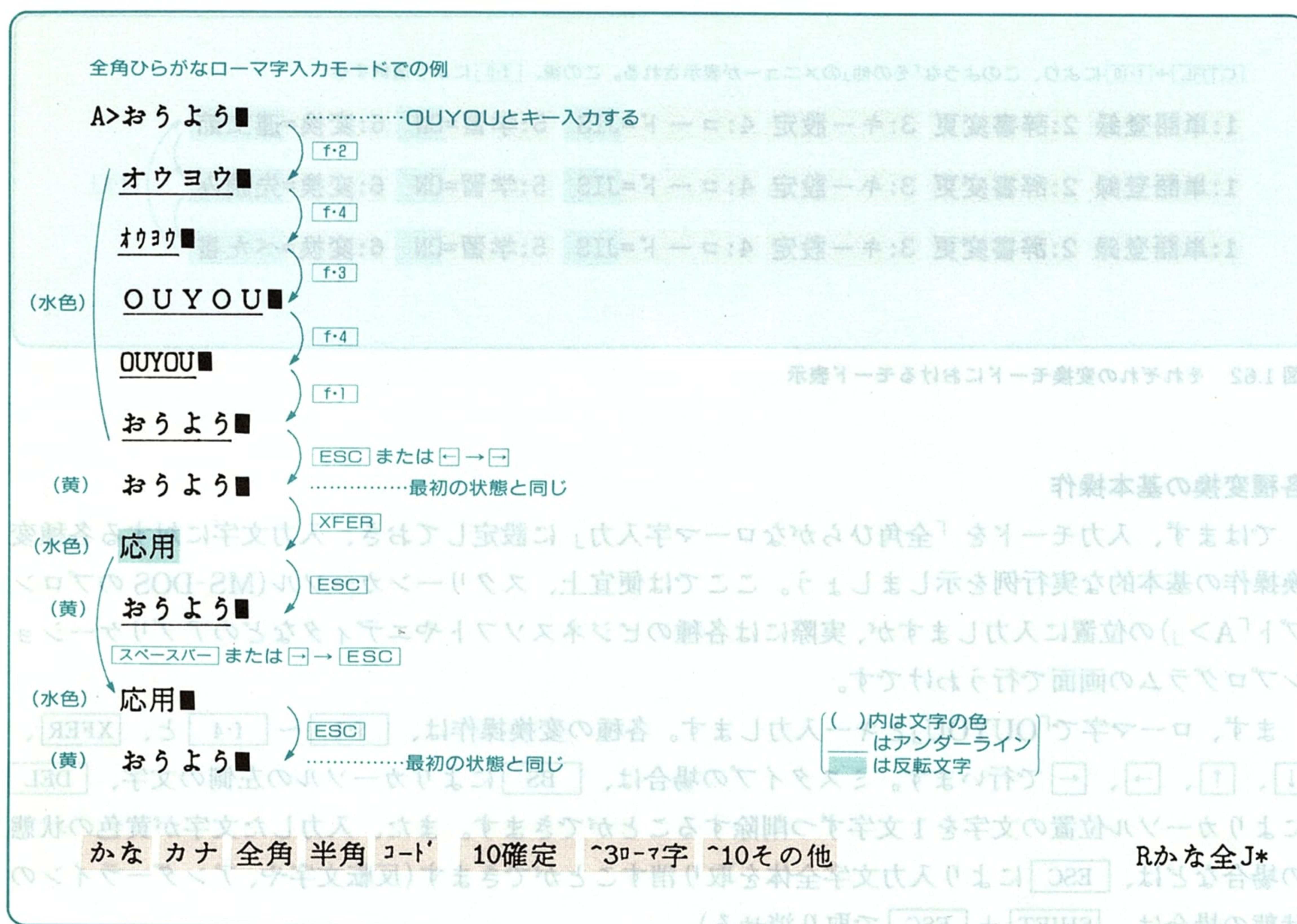



図 1.63　各種変換の基本操作

かな－漢字変換

漢字や熟語などの登録語への変換(いわゆる、かな－漢字変換)は、任意の入力モードで「読み」を入力した後、XFER の入力により行われます。変換された語が目的のものでなければ、再度 XFER を入力することにより次の登録語が表示されます。次々と XFER を入力しながら１語ずつ捜すこともできますが、XFER の代わりに ↓ を入力した場合は、メニュー行に変換候補群が表示されますので、このなかに目的の語が存在すれば、→、← や、XFER、あるいは SHIFT ＋ XFER を使って、目的の語にカーソルを合わせて選択するか、またはその番号を入力して目的の語を取り出します。取り出した語の固定は、前述のとおりです。もし、その変換候補群に目的の語が見つからない場合は、さらに ↓ を入力し、次の変換候補群のなかから捜します。１つ前の候補群にもどるには ↑ を入力します。

ではこの実行例を示しましょう。スクリーンモードの場合もメニューラインモードの場合も、これらの変換の手順は同じです。

A>しょ■ ……………SYOとキー入力

かな　カナ　全角　半角　ｺｰﾄﾞ　10確定　^3ﾛｰﾏ字　^10その他　　Rかな全J*

XFER

A>処

かな　カナ　全角　半角　ｺｰﾄﾞ　10確定　^3ﾛｰﾏ字　^10その他　　Rかな全J*

↓を入力して候補群を表示した後、→、←で語を選択する

A>書

選択　1:処　2:初　3:所　4:暑　5:書　6:渚　7:署　8:緒　9:曙

選択　1:薯　2:藷　3:諸　4:庶　5:黍　6:岨　7:しょ

↓により次の候補群が表示される。
↑により前の候補群にもどる。
あるいは→、←を続けて入力しても同様

かな　カナ　全角　半角　ｺｰﾄﾞ　10確定　^3ﾛｰﾏ字　^10その他　　Rかな全J*

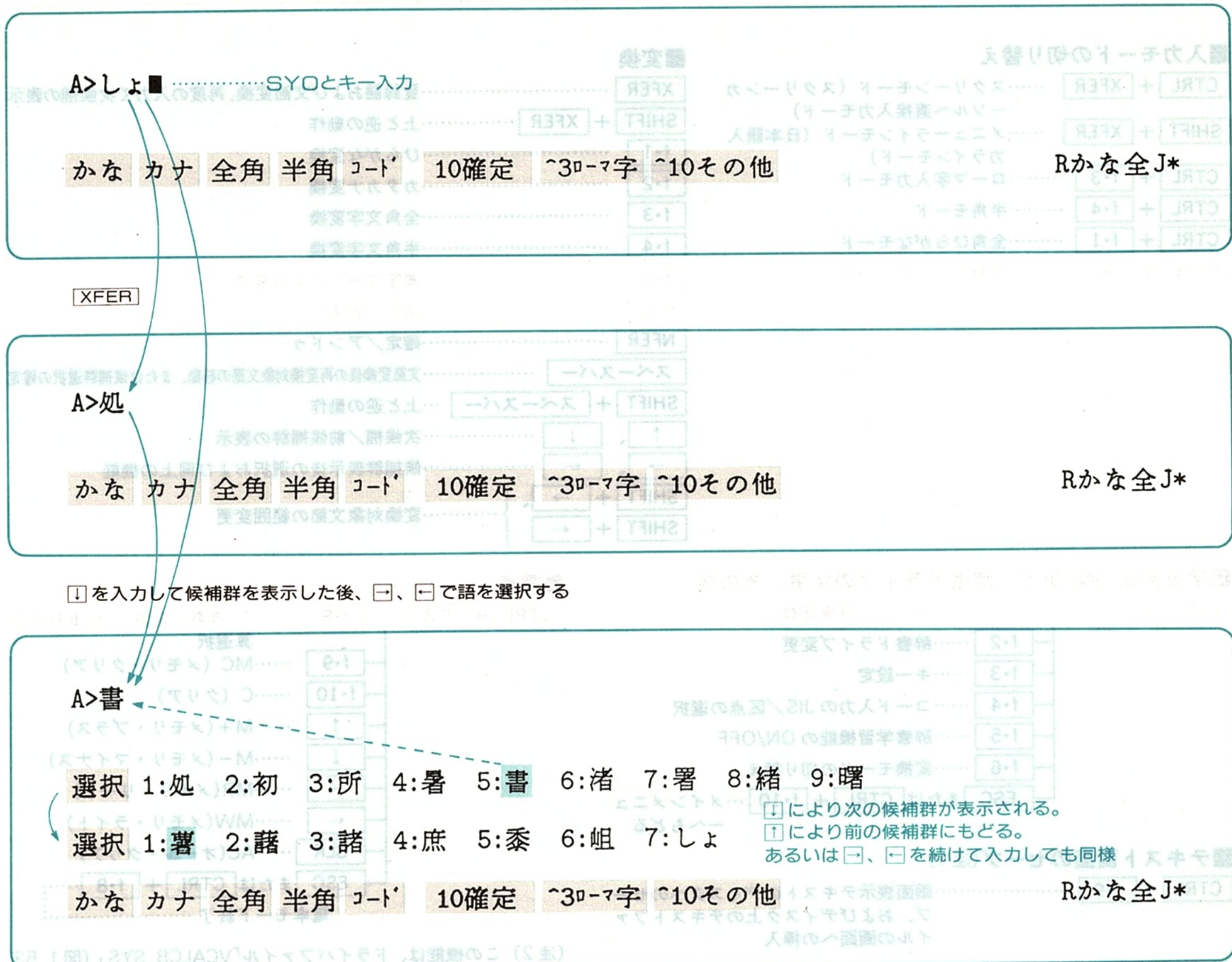



図 1.64　漢字・熟語などの登録語への変換(かな－漢字変換)

ここでVJE-βの基本操作に関する各種キーの機能と、ローマ字入力の綴り方についてまとめておきましょう。

■入力モードの切り替え

CTRL ＋ XFER ……スクリーンモード（スクリーンカーソルへ直接入力モード）
SHIFT ＋ XFER ……メニューラインモード（日本語入力ラインモード）
CTRL ＋ f･3 ………ローマ字入力モード
CTRL ＋ f･4 ………半角モード
CTRL ＋ f･1 ………全角ひらがなモード
CTRL ＋ f･2 ………全角カタカナモード

■変換

XFER ……………登録語および文節変換、再度の入力で次候補の表示
SHIFT ＋ XFER ……………上と逆の動作
f･1 ……………ひらがな変換
f･2 ……………カタカナ変換
f･3 ……………全角文字変換
f･4 ……………半角文字変換
f･5 ……………漢字コードによる変換
f･10 ……………確定（固定）
NFER ……………確定／アンドゥ
スペースバー ……………文節変換後の再変換対象文節の移動、または候補群選択の確定
SHIFT ＋ スペースバー …上と逆の動作
↑、↓ ……………次候補／前候補群の表示
→、← ……………候補群表示後の選択および同上の機能
SHIFT ＋ →、SHIFT ＋ ← ……………変換対象文節の範囲変更

■学習機能 ON/OFF、辞書ドライブの変更、その他

CTRL ＋ f･10 ─
- f･1 ……ユーザー辞書登録
- f･2 ……辞書ドライブ変更
- f･3 ……キー設定
- f･4 ……コード入力の JIS／区点の選択
- f･5 ……辞書学習機能の ON/OFF
- f･6 ……変換モードの切り替え
- ESC または CTRL ＋ f･10 …メインメニューへもどる

■テキスト画面のセーブ(注 1)

CTRL ＋ f･9 ……………画面表示テキストのディスクへのセーブ、およびディスク上のテキストファイルの画面への挿入

（注 1）この機能は、ドライバファイル「VFILEB.SYS」（図 1.53 参照）がCONFIG.SYS ファイルに「DEVICE＝VFILEB.SYS」として登録してある場合に動作する。

■電卓(注 2)

CTRL ＋ f･8 ─
- f･8 ……10 進数の計算／16 進数の計算選択
- f･9 ……MC（メモリ・クリア）
- f･10 ……C（クリア）
- ↑ ……M＋(メモリ・プラス)
- ↓ ……M－(メモリ・マイナス)
- → ……MR(メモリ・リード)
- ← ……MW(メモリ・ライト)
- CLR ……AC(オール・クリア)
- ESC または CTRL ＋ f･8 ……電卓モード終了

（注 2）この機能は、ドライバファイル「VCALCB.SYS」（図 1.53 参照）が CONFIG.SYS ファイルに「DEVICE＝VCALCB.SYS」として登録してある場合に動作する。

図 1.65　各種キーの機能

長　音	
ローマ字	RO－MAZI*
ユーザー	YU－ZA－
つまる音	
がっこう(学校)	GAKKOU
いっち(一致)	ITTI
あっ	AXTU
うっ	UXTU

＊－は「マイナス」または「ハイフン」記号

を、ん	
を	WO
ん	NN または N′
しんい(真意)	SINNI または SIN′I
ほんしゃ(本社)	HONNSYAまたはHON′SYA
かな小文字	
ぁ	XA
ぃ	XI
ゅ	XYU

あ	あ A	い I	う U	え E	お O	ぁ XA	ぃ XI	ぅ XU	ぇ XE	ぉ XO
か	か KA XKA	き KI	く KU	け KE XKE	こ KO	きゃ KYA	きぃ KYI	きゅ KYU	きぇ KYE	きょ KYO
さ	さ SA	し SI SHI	す SU	せ SE	そ SO	しゃ SYA SHA	しぃ SYI	しゅ SYU SHU	しぇ SYE SHE	しょ SYO SHO
た	た TA	ち TI CHI	つ TU TSU っ XTU	て TE	と TO	ちゃ TYA CYA CHA てゃ THA	ちぃ TYI CYI てぃ THI	ちゅ TYU CYU CHU てゅ THU	ちぇ TYE CYE CHE てぇ THE	ちょ TYO CYO CHO てょ THO
な	な NA	に NI	ぬ NU	ね NE	の NO	にゃ NYA	にぃ NYI	にゅ NYU	にぇ NYE	にょ NYO
は	は HA	ひ HI	ふ HU FU	へ HE	ほ HO	ひゃ HYA ふぁ FA ふゃ FYA	ひぃ HYI ふぃ FI FYI	ひゅ HYU ふ FU ふゅ FYU	ひぇ HYE ふぇ FE FYE	ひょ HYO ふぉ FO ふょ FYO
ま	ま MA	み MI	む MU	め ME	も MO	みゃ MYA	みぃ MYI	みゅ MYU	みぇ MYE	みょ MYO
や	や YA	い YI	ゆ YU	え YE	よ YO	ゃ XYA	ぃ XYI	ゅ XYU	ぇ XYE	ょ XYO
ら	ら RA	り RI	る RU	れ RE	ろ RO	りゃ RYA	りぃ RYI	りゅ RYU	りぇ RYE	りょ RYO
わ	わ WA XWA	うぃ WI	う WU	うぇ WE	を WO					
ん	ん NN N′									
が	が GA	ぎ GI	ぐ GU	げ GE	ご GO	ぎゃ GYA	ぎぃ GYI	ぎゅ GYU	ぎぇ GYE	ぎょ GYO
ざ	ざ ZA	じ ZI JI	ず ZU	ぜ ZE	ぞ ZO	じゃ ZYA JA JYA	じぃ ZYI JYI	じゅ ZYU JU JYU	じぇ ZYE JE JYE	じょ ZYO JO JYO
だ	だ DA	ぢ DI	づ DU	で DE	ど DO	ぢゃ DYA でゃ DHA	ぢぃ DYI でぃ DHI	ぢゅ DYU でゅ DHU	ぢぇ DYE でぇ DHE	ぢょ DYO でょ DHO
ば	ば BA	び BI	ぶ BU	べ BE	ぼ BO	びゃ BYA	びぃ BYI	びゅ BYU	びぇ BYE	びょ BYO
ぱ	ぱ PA	ぴ PI	ぷ PU	ぺ PE	ぽ PO	ぴゃ PYA	ぴぃ PYI	ぴゅ PYU	ぴぇ PYE	ぴょ PYO
ヴ	ヴァ VA	ヴィ VI	ヴ VU	ヴェ VE	ヴォ VO					

（この表は主要なものだけで、その他の綴り方は省略します）

表 1.5　VJE-β におけるローマ字の綴り方

■ 文節変換

VJE-βにおける連文節変換の実際を、次の文(ATOK6や松茸の例と同じ)を入力する場合を例に解説しましょう。

1969年7月21日，宇宙船アポロ11号のアームストロング飛行士らは，
月への着陸に成功し，その表面に初めて人類の足跡を残した.

この文章を入力するには、文節のいろいろな区切り方がありますが、この区切り方は作業能率に大きく関係しますので、経験上から適当な区切り方を修得してください。すべてのFEPや、ワープロ専用機についても言えることですが、あまり長い文を一度に変換することはお勧めできません。短く区切って確実に変換していくことが、結局は能率的なのです。しかしここでは、次ページの図1.66のように、わざと長く区切った例として連文節変換を行ってみましょう。ローマ字入力の綴り方は、36ページの図1.25や60ページの図1.48のATOK6や松茸での例と同じです。

連文節変換が行われると、変換された文字列が、水色と水色反転部とに色分けされます(文の最初の文節が水色反転、そのほかはすべて水色)。全体が目的どおりに変換されていれば、次の「読み」の入力や、[f·10]、[NFER]、または[⏎]の入力などで全体を確定します。もし、どこかの文節が間違って変換されていれば、[→]、[←]により、水色反転部をその文節に移して、その部分に対して[XFER]または[↓]による再変換を行います。

ここで連文節変換の一部が、目的どおりに変換されなかった場合の修正処理の基本について簡単に示しておきましょう。

- 現在変換されている文の水色反転部は、[XFER]、[↓]、[f·1]～[f·4]などによる再変換が可能
- 水色反転部は、VJE-βが判別した文節の区切りであり、[←]、[→]により任意の文節に水色反転部を移動できる
- 文節の区切り方が誤っている場合は、水色反転部をその文節に移動し、[SHIFT]＋[←]や[SHIFT]＋[→]で1文字ずつ前後に区切りを変更できる
- 水色反転部は[ESC]により、入力時の「読み」の状態に開かれる(もどる)
- 黄色および黄色＋アンダーラインの部分は、常に再変換可能である
- 黄色の部分は[ESC]によりその全部が削除される

ローマ字の綴りは図1.25と同じ

１９６９ねん７がつ２１にち，
XFER
１９６９年７月２１日，

うちゅうせんあぽろ１１ごうの
XFER
宇宙船アポロ１１合の
（水色）
→→→ ……………再変換の対象となる文節の水色反転部を移動する
宇宙船アポロ１１合の
（水色）
XFER ……………再変換
宇宙船アポロ１１号の
（水色）

あーむすとろんぐひこうしらは，
XFER
アームストロング飛行白羽，
（水色）
→→→
アームストロング飛行白羽，
（水色）
ESC
アームストロングひこう白羽，
（黄）
SHIFT＋→ ……………文節の区切りを SHIFT＋→／← によって変更する
アームストロングひこうしらは，
（黄）
XFER ……………再変換
アームストロング飛行士らは，
（水色）

つきへのちゃくりくにせいこうし，
XFER
月への着陸に成功し，

そのひょうめんにはじめて
XFER
その表面に初めて

じんるいのそくせきをのこした．
XFER
人類の即席を残した．
（水色）
→
人類の即席を残した．
（水色）
XFER ……………再変換
人類の足跡を残した．
確定

図 1.66　連文節変換による実際のキー入力の一例

■ JIS漢字コードによる漢字／記号等の入力

JIS漢字コード、または区点コードにより、漢字や記号などを入力することができます。どちらのコードを使うかは、[CTRL]＋[f･10]により「その他」のメニューを選択した後、[f･4]によって、JIS／区点の各コードの選択ができます。また、後述するCONFIG.SYSファイルへの登録によっても、起動時の初期設定として、どちらかのコードを設定できます。

では、JIS漢字コードによる変換の一例を示しましょう。

「3D71」を入力して、[f･5]を押すとこのような画面になる

```
A>3 D 7 1
```

[↵]を押すと、現在選択されている文字がスクリーンカーソル上に表示される

[→]、[←]、[↓]、[↑]によって選択する

```
選択 1:書  2:薯  3:藷  4:諸  5:助  6:叙  7:女  8:序  9:徐  [3D71]      Rかな全J*
```

選択されている文字のJISコードの表示

JISコードのモードを示す（区点モードの場合は「K」）

目的の文字の取り出しは、[←]、[→]、[↓]、[↑]で選択した後、[f･10]、[↵]、[スペースバー]などで確定する。また、数字を入力することによっても確定する

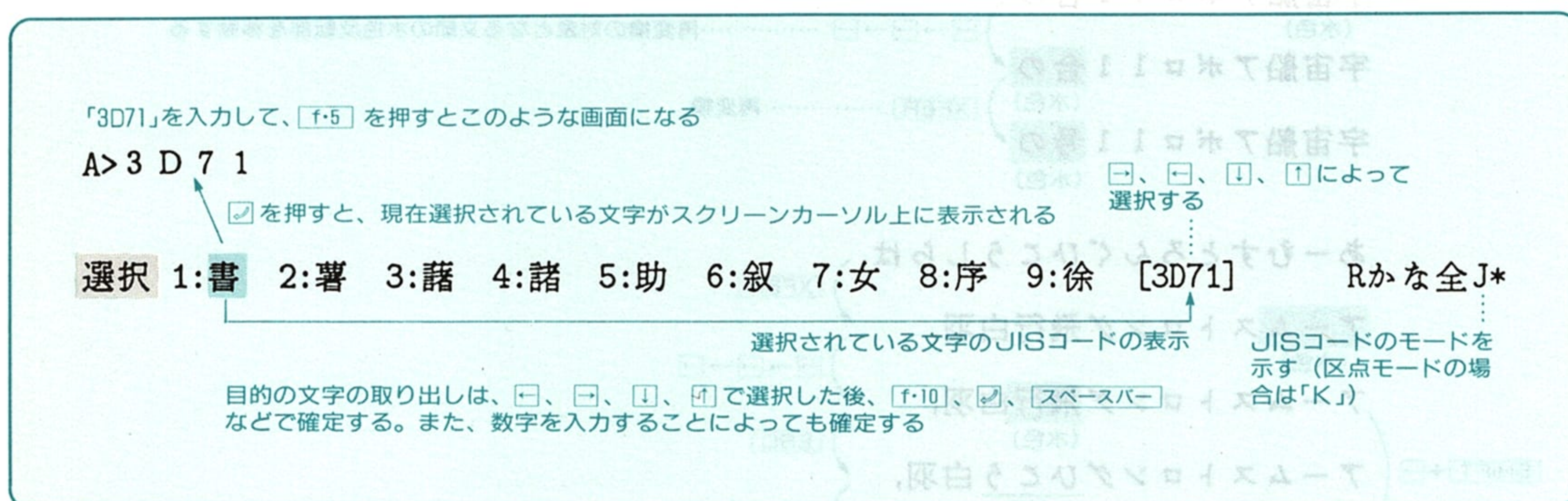



図1.67　JIS漢字コードによる変換

■ 辞書登録語の追加／変更機能

VJE-βの辞書の登録語の追加や削除には、辞書ファイル保守のための辞書登録修正ユーティリティプログラム「VEDIT.EXE」（図1.53参照）が用意されていますが、登録に関しては日本語入力モードから直接行うことができます。モードは、スクリーンモード、メニューラインモードのいずれの場合でも可能です。

では実行例として、日本語入力モードの画面上から、「ワードプロセッサ」という語を、「わぷ」という「読み」で登録してみましょう。まず、[CTRL]＋[f･10]で「そのほかの機能メニュー」を表示した後、[f･1]を入力します。

(画面上の任意の位置にこの語があるとする)

A>.....ワードプロセッサ.....

どこから?

どこまで?

←、→、↓、↑を使って指定した後、⏎する

読み　わぷ⏎　……………読みを入力

数字を入力し⏎する

品詞　1:一般名詞　2:人名　3:地名　4:固有名詞　5:サ変化名詞　6:単漢字　:1⏎

登録が完了し、自動的にメインメニュー画面にもどる。作業の途中からは、
ESC によりいつでもメインメニューにもどれる

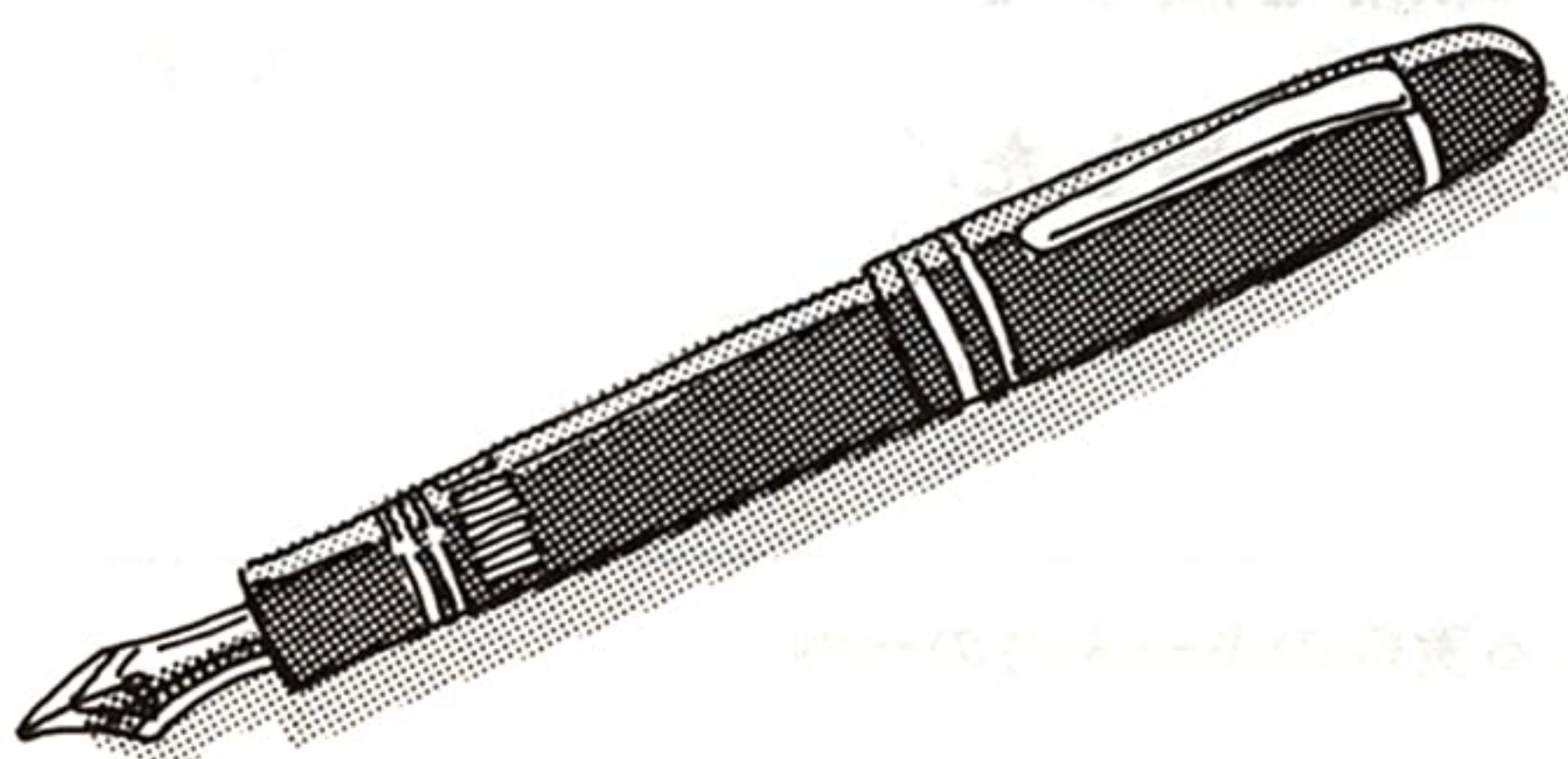
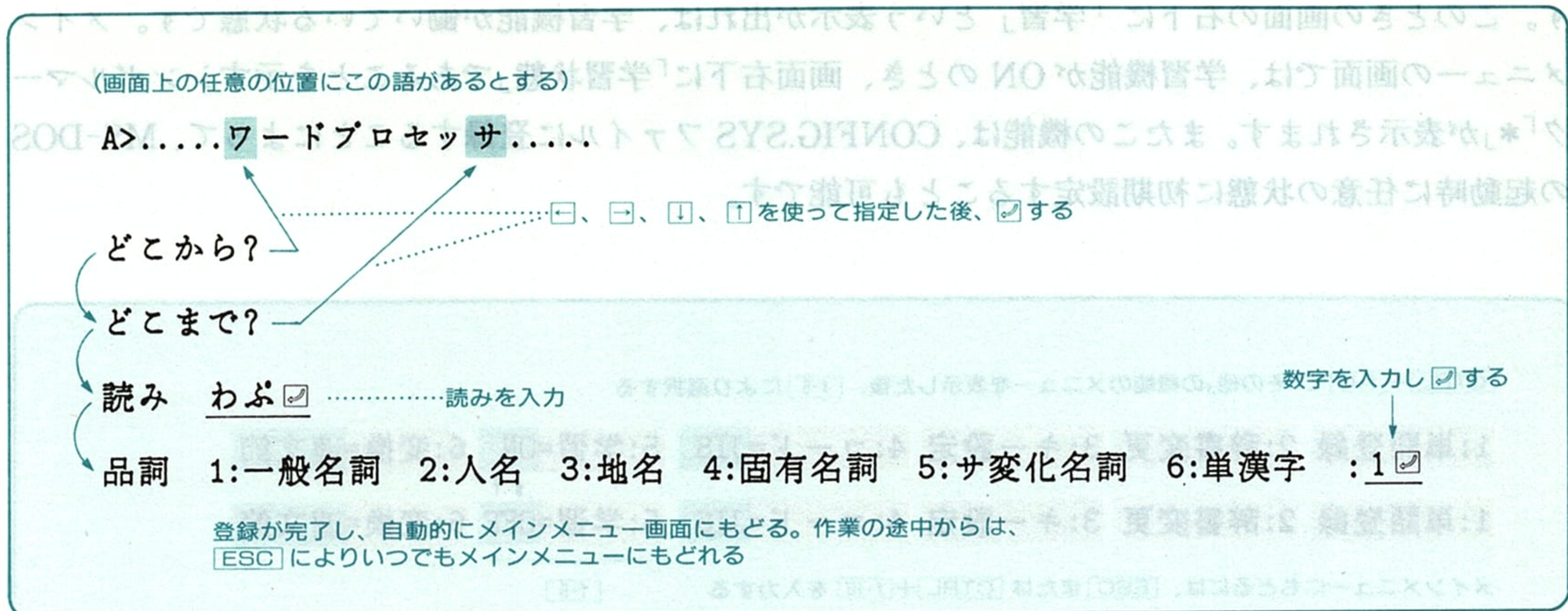



図 1.68　日本語入力モードから辞書に新しい語を登録する

以上の操作で新しい語の登録が終わりました。登録語は、VJE-β の辞書ファイル「VJEB.DIC」に追加されました。確認のために「わぷ」と入力して、XFER で変換してみましょう。

A>わぷ

XFER

ワードプロセッサ ……………「わぷ」と入力し、XFER により「ワードプロセッサ」に変換された!

かな　カナ　全角　半角　コード　10確定　^3ローマ字　^10その他　　Rかな全J*

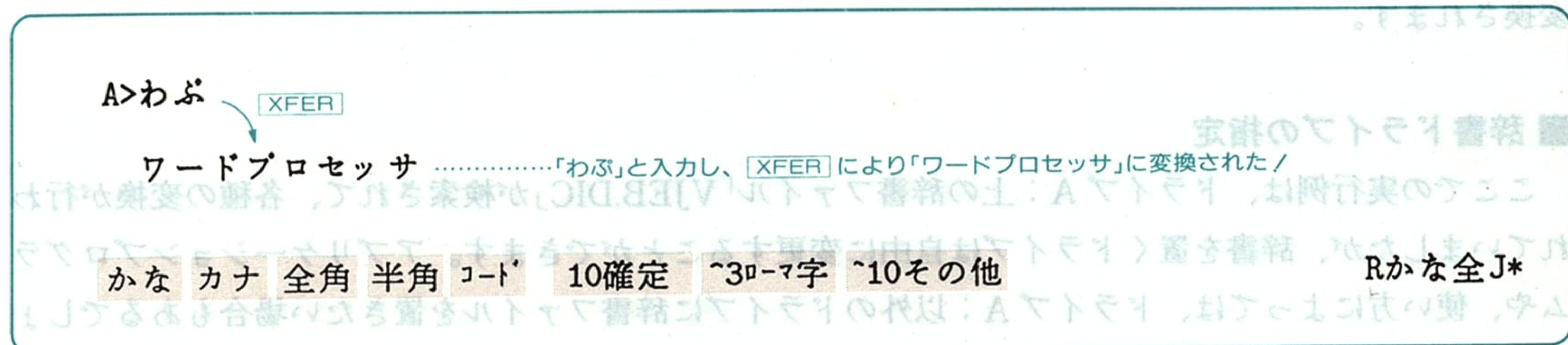



図 1.69　新しく登録した語の変換の確認

辞書登録修正ユーティリティプログラム「VEDIT.EXE」を使うと、辞書への登録だけでなく、登録後の削除や並べ替えなどを行うことができます。このプログラムの操作法は、メニューによる対話方式ですので、起動すれば(MS-DOS において「A>VEDIT ⏎」)使い方は自然にわかるようになっています。

■ 学習機能の ON／OFF

VJE-β では、かな－漢字変換や単字変換などの登録語への変換の際、同じ読みの登録語が複数存在する場合には、もっとも最近に使われた登録語の順に取り出されて変換されます。これは辞書に学習機能があるからですが、その機能は必要があれば OFF することができます。この ON／OFF の切り替えは、CTRL ＋ f·10 を入力して、「そのほかの機能メニュー」を表示した後、f·5 により行いま

す。このときの画面の右下に「学習」という表示が出れば、学習機能が働いている状態です。メインメニューの画面では、学習機能がONのとき、画面右下に「学習状態」であることを示すシンボルマーク「*」が表示されます。またこの機能は、CONFIG.SYSファイルに登録することによって、MS-DOSの起動時に任意の状態に初期設定することも可能です。

CTRL＋f·10で「その他」の機能のメニューを表示した後、f·5により選択する

```
1:単語登録 2:辞書変更 3:キー設定 4:コード=JIS 5:学習=ON  6:変換=連文節
1:単語登録 2:辞書変更 3:キー設定 4:コード=JIS 5:学習=OFF 6:変換=連文節
```

メインメニューにもどるには、ESCまたはCTRL＋f·10を入力する　f·5

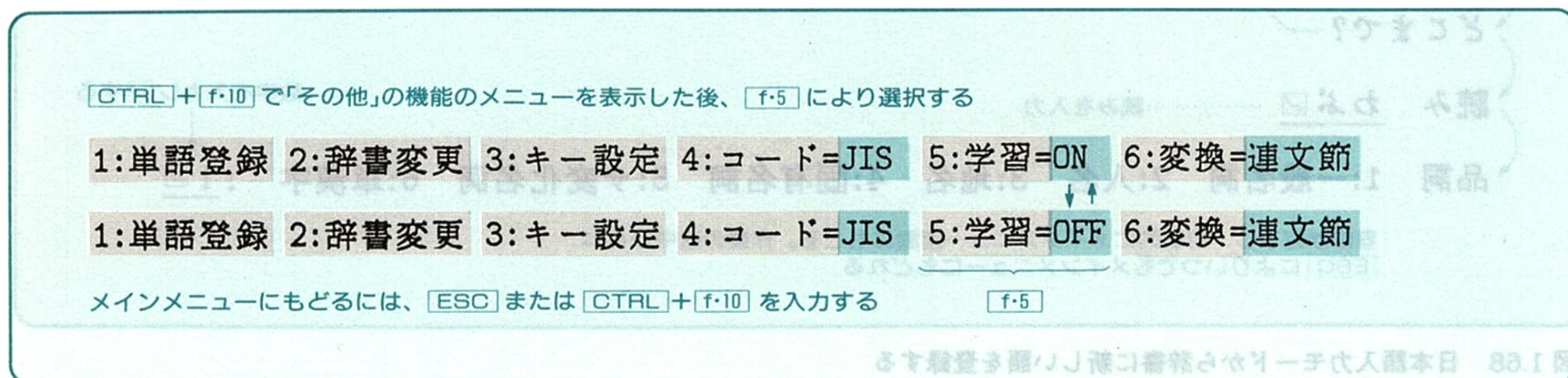



図 1.70　学習機能の ON／OFF の選択

学習機能がONの状態では、登録語への変換が行われ、それが確定されるたびに、ディスク上の辞書ファイルの優先順位が書き換えられます。これにより、次回の変換は、過去最新に使った語の順に変換されます。

■ 辞書ドライブの指定

ここでの実行例は、ドライブA：上の辞書ファイル「VJEB.DIC」が検索されて、各種の変換が行われていましたが、辞書を置くドライブは自由に変更することができます。アプリケーションプログラムや、使い方によっては、ドライブA：以外のドライブに辞書ファイルを置きたい場合もあるでしょう。このためVJEには、辞書ドライブを変更する機能があり、日本語入力モードにおいてCTRL＋f·10を入力した後、f·2（辞書変更）で任意のドライブを自由に指定することができます。ただし、いずれの場合も、辞書ファイルはルートディレクトリに置かなければなりません。

```
A>■
```

CTRL＋f·10で「その他」の機能のメニューを表示した後、f·2により下のメッセージが表示される

```
辞書ドライブ変更 ドライブ名を指定して下さい[A,B ...] B⏎
```

これ以降、ドライブB：上の辞書が使用される

任意のドライブ名を入力して⏎する。指定が完了するとメインメニューにもどる

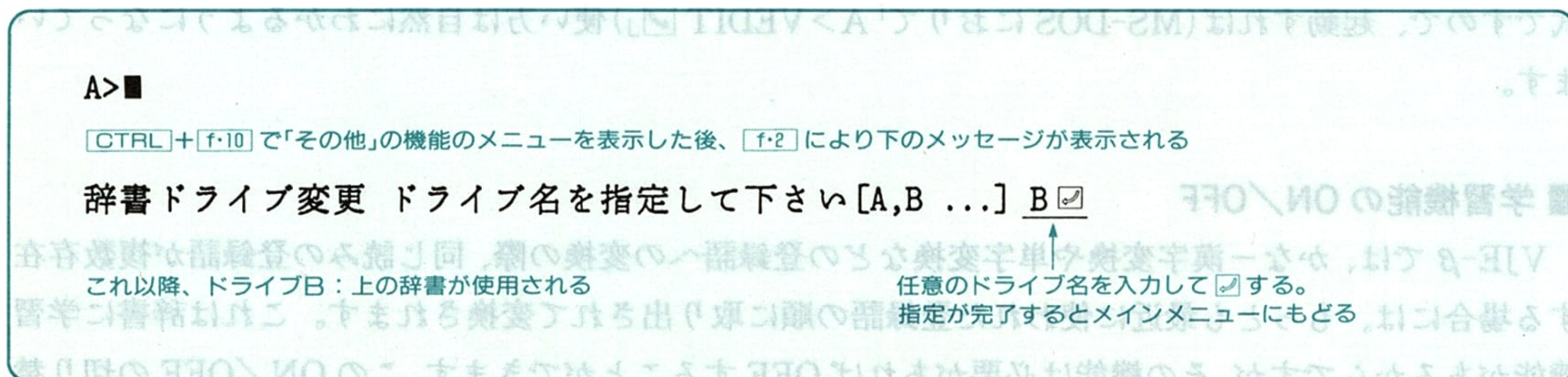



図 1.71　辞書を置くドライブの変更

■ MS-DOS 起動時の VJE-β の初期設定

CONFIG.SYS ファイルに登録する VJE-β のデバイスドライバ(図 1.56 参照)に、各種のパラメータを付けることにより、VJE-β の各種のモードを任意に初期設定することができます。そのパラメータの一覧表を次ページの**表 1.6** に示します。

これらのパラメータは、スペースで区切って記述することにより、複数の項目を指定することができます。その一例を示します。

例：DEVICE＝VJEB.DRV_/B：_/M1_/HR_/J_/L_/SYS＝B：¥SYS

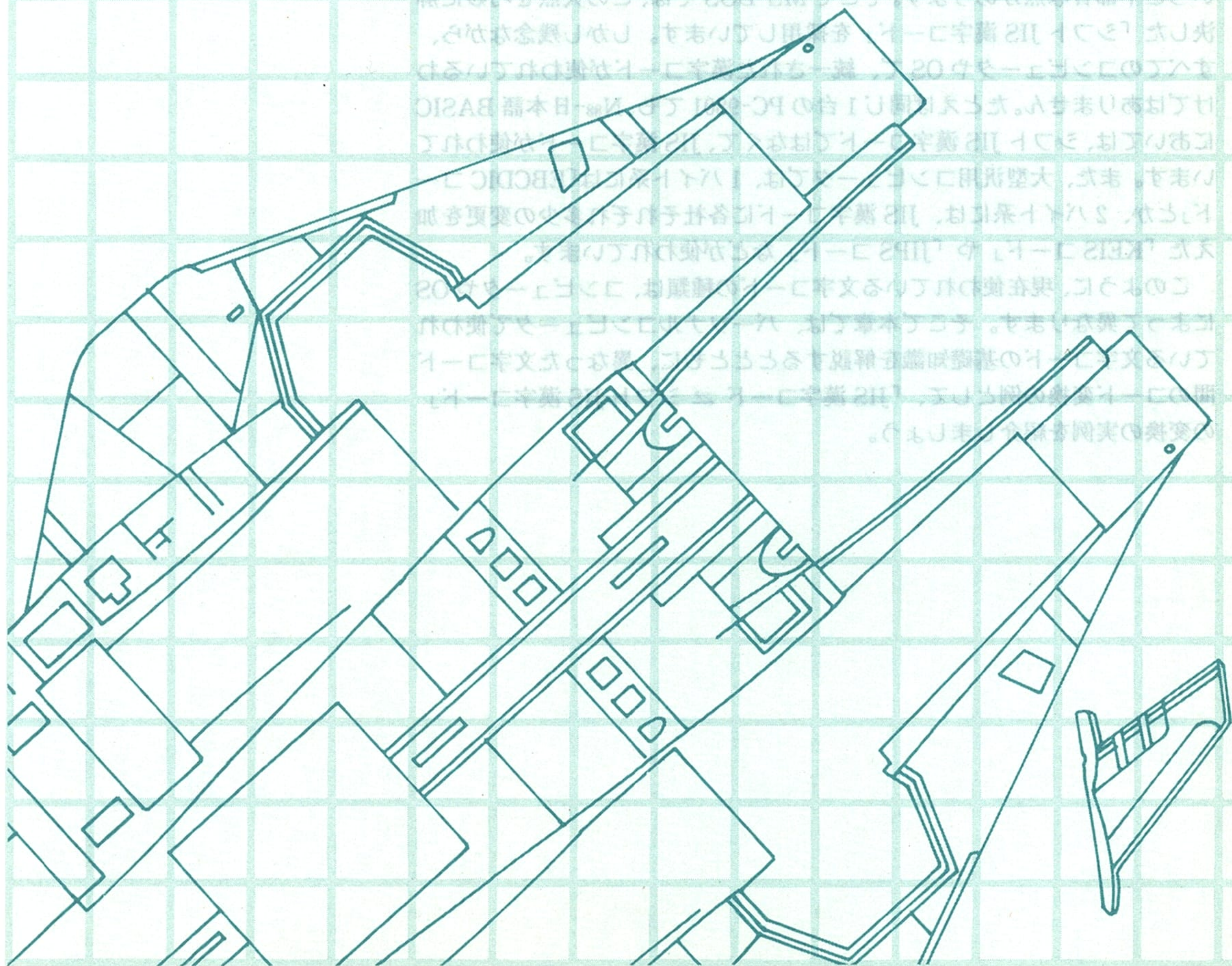
(「_」はスペースを表す)

- /B： … 辞書ドライブはB：
- /M1 … 全角ひらがな(ローマ字入力モード)
- /HR … 連文節変換モード
- /J … JIS 漢字コードによる入力モード
- /L … 辞書書き換え学習モード
- /SYS＝B：¥SYS … 「VJEB.SYS」を置くディレクトリを指定

また、これらのパラメータは、「/ *x*：」、「/T*x*」、「/K*x*」、「/SYS」、「/D」、「/G」、「/V*x*」を除いて、VJE-β 初期状態設定ユーティリティプログラム「SETVJE.EXE」を実行することにより、CONFIG.SYS ファイルに自動的に書き込まれます。このユーティリティプログラムは、メニューによる対話形式になっていますので実行は簡単です。各自で試みてください。

パラメータ	機　能
/x:	辞書を置くドライブの設定
/L	辞書学習モード（無指定:学習機能 OFF）
/J	JIS コード入力（無指定:区点コード入力）
/Mn	入力初期モード設定 0　半角カタカナ（カナキー）／半角英数 1　全角ひらがな（ローマ字） 2　全角ひらがな（カナキー）／全角英数 3　全角カタカナ（ローマ字） 4　全角カタカナ（カナキー）／全角英数 5　半角カタカナ（ローマ字）
/H$	入力初期モード設定。$には以下の文字を指定 B　べた書き入力 S　先読み入力 R　連文節変換
/B	白黒モニタや液晶ディスプレイ使用時に設定 （無指定:カラー）
/Tn	キーセンス・アイドル回数（キー入力の待ち時間）の設定。数が少ないほど、カーソル移動などが速くなる（T 指定のない場合は 100 回が初期値） 0　4回 1　10回 2　20回 3　30回 ⋮　⋮ 9　100回
/N	スクリーンモードを禁止する
/S	べた書き入力での自動確定を禁止する
/SYS=	＝のあとに指定したディレクトリにある VJEB.SYS を起動
/Dn	選択モードに移行するまでの変換キー入力回数の設定 0　選択モードへの移行を禁止 1　1回 2　2回 3　3回 ⋮　⋮ 9　9回
/K$	キー入力初期モード設定。$には以下の文字を指定 S　スペースバーを変換キーに設定 R　リターンキーを確定キーに設定 G　[GRPH]によるローマ字入力モードの ON/OFF を設定 、　[、]でカンマ（,）を入力 。　[。]でピリオド（.）を入力 ,　[,]で読点（、）を入力 .　[.]で句点（。）を入力 ー　[ー]で音引（ー）を入力 [　[[][]]でカギカッコ（「」）を入力 X　先読み変換の利用時に、句読点入力ごとに変換を実行

表 1.6　VJE-β の初期設定用パラメーター覧表

2章
漢字コードの種類と変換

パーソナルコンピュータには、主に次の3種類の文字コードが使われています。

- ●アスキーコード
- ● JIS 漢字コード
- ●シフト JIS 漢字コード

「アスキーコード」は、1文字を1バイトで表現するもので、「1バイト系コード」と呼ばれ、ほとんどすべてのパーソナルコンピュータの基本文字コードとして使われています。

しかし、1バイト(8ビット)で表せる文字の種類は256以下であり、何千種類とある漢字を表すことはできません。そこで日本では、1文字を2バイト(16ビット)で表現する「2バイト系コード」が規格化され、「JIS 漢字コード」として制定されています。

ところが、この JIS 漢字コードには、コンピュータで処理する上でいろいろと不都合な点があります。そこで MS-DOS では、この欠点を巧妙に解決した「シフト JIS 漢字コード」を採用しています。しかし残念ながら、すべてのコンピュータや OS で、統一された漢字コードが使われているわけではありません。たとえば同じ1台の PC-9801 でも、N$_{88}$-日本語 BASIC においては、シフト JIS 漢字コードではなくて、JIS 漢字コードが使われています。また、大型汎用コンピュータでは、1バイト系には「EBCDIC コード」とか、2バイト系には、JIS 漢字コードに各社それぞれ多少の変更を加えた「KEIS コード」や「JIPS コード」などが使われています。

このように、現在使われている文字コードの種類は、コンピュータや OS によって異なります。そこで本章では、パーソナルコンピュータで使われている文字コードの基礎知識を解説するとともに、異なった文字コード間のコード変換の例として、「JIS 漢字コード ⇄ シフト JIS 漢字コード」の変換の実例を紹介しましょう。

2.1 アスキーコード

ASCII (American Standard Code for Information Interchange) コードは、多くのコンピュータに広く使われている代表的な文字コードであり、とくにパーソナルコンピュータの世界では、MSX マシンから、UNIX マシンなどのワークステーションと呼ばれている高性能マシンまで、ほとんどすべてに使われています。

アスキーコードは、1文字を1バイトの値(コード)で表します。「バイト」というのは、コンピュータの内部処理における数値の最小単位であり、8ビットで構成されています。8ビット(1バイト)で表せる数値の範囲は、16進数の 0〜FF_H(10進数の 0〜255)であり、つまり 256 種類の文字を表すことが可能です。

このように、1バイトの各数値に1文字を割り当てて作られたアスキーコード表(オリジナルのアスキーコードにカタカナを組み入れた日本版アスキーコード表)が、巻末の APPENDIX にありますので参照してください。その表から、いくつかの文字のコードを拾い出してみましょう。

A … 41_H　　a … 61_H　　Z … $5A_H$
\# … 23_H　　(… 28_H　　* … $2A_H$　　($_H$は 16 進数を表す)

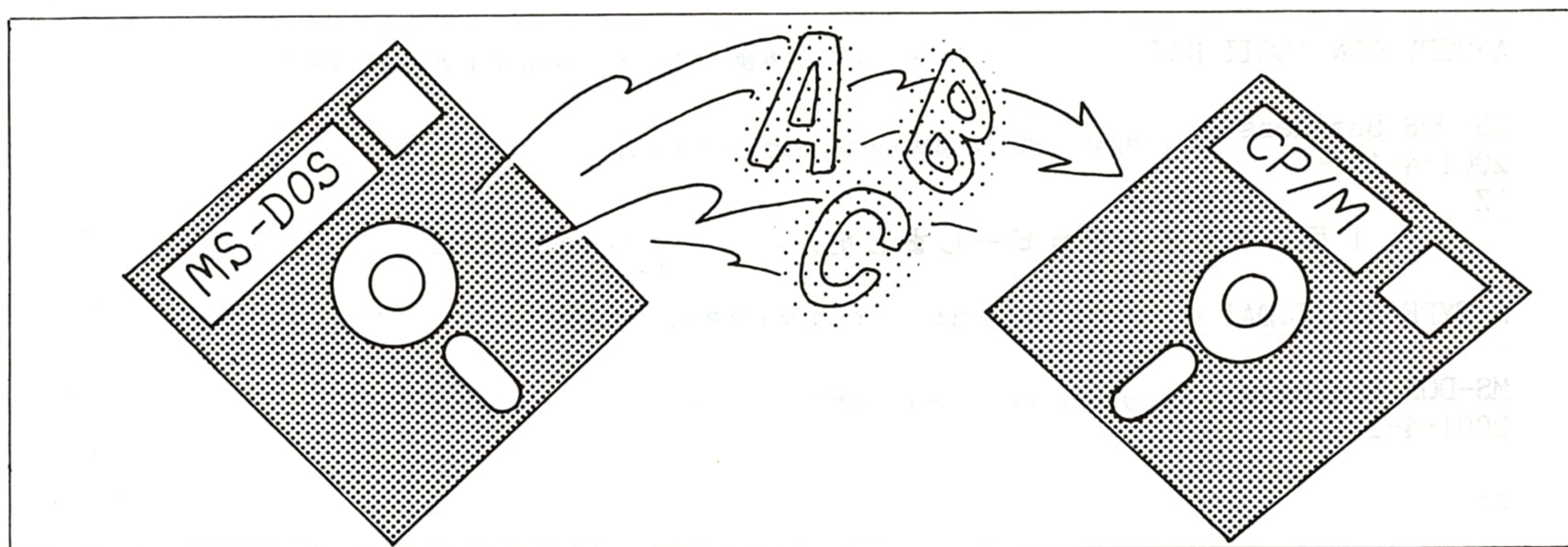
MS-DOS、OS/2、MSX、CP/M、BASIC などを基本ソフトウェアとする、ほとんどすべてのパーソナルコンピュータの操作は、アスキーコードによるコマンドで行われます。たとえば、MS-DOS の「DIR コマンド」を実行する際、「DIR↵」とキー入力しますが、この入力文字はアスキーコードでなければなりません。つまり、通常入力による半角文字であり、FEP(日本語入力システム)の日本語入力による全角文字で「ＤＩＲ↵」とキー入力しても、DIR コマンドは実行されません。

```
A>DIR *.SYS/W⏎ ……………通常の入力モードでDIRコマンドを実行する(半角文字)

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

EMSDRIVE SYS     GRAPH    SYS     MOUSE    SYS     PRINT    SYS     RAMDISK  SYS
RSDRV    SYS     CONFIG   SYS
        7 個のファイルがあります.
    26624 バイトが使用可能です.

A>ＤＩＲ　＊．ＳＹＳ／Ｗ⏎ ……………日本語入力モードの全角文字でDIRコマンドを実行する
コマンドまたはファイル名が違います. ……………DIRコマンドは実行されない

A>

_                                   日本語入力モード …………… 連ローマ字漢字
```

図 2.1　MS-DOS コマンドはアスキーコードの文字入力で実行される

ではここで、簡単な文字ファイルを作成して、アスキーコードを具体的に見てみましょう。

```
MS-DOS Business
2001-4-1.
```

このような簡単な文字ファイルですので、エディタプログラムを使わずに COPY コマンドで作成しましょう(くわしくは 5.1 章参照)。キー入力は、日本語入力モードによる全角文字ではなく、通常入力モードで行います。その実行例を次に示します。

```
A>COPY CON ASCII.DAT⏎ ……………COPYコマンドを使ってファイル「ASCII.DAT」を作成する

MS-DOS Business⏎ ┐
2001-4-1.⏎       ┘内容はこの2行。通常入力モードでキー入力する
^Z⏎ ……………CTRL+Z ⏎
         1 個のファイルをコピーしました. ……………ファイルが作成された

A>TYPE ASCII.DAT⏎ ……………作成されたファイルをタイプアウトして確認

MS-DOS Business ┐
2001-4-1.       ┘タイプアウトされたファイルの内容

A>
```

図 2.2　簡単な文字ファイルを COPY コマンドで作成する

このように、たった2行の文字ファイル **ASCII.DAT** が作成されました。キー入力は通常入力で行っていますので、ディスク上にはアスキーコードの文字列が格納されているはずです。これを DUMP プログラムで確認してみましょう。DUMP プログラムは、MS-DOS のシステムディスクに **DUMP.EXE**（あるいは.COM）というファイル名で付属しているプログラムであり、各種のファイルのデータの内容を、1バイトごとに16進数で表示(ダンプ)します。

```
A>DUMP ASCII.DAT

 Dump Version 3.00
          M  S  -  D  O  S  ␣  B  u  s  i  n  e  s  s  復帰コード
00000000  4D 53 2D 44 4F 53 20 42-75 73 69 6E 65 73 73 0D   MS-DOS Business.
00000010  0A 32 30 30 31 2D 34 2D-31 2E 0D 0A 1A            .2001-4-1....
    改行コード 2  0  0  1  -  4  -  1  .  復帰 改行 ファイルの終わりを示すコード
A>TYPE ASCII.DAT
MS-DOS Business
2001-4-1.

A>
```

…………作成したファイル「ASCII.DAT」の内容データをDUMPプログラムで見る

TYPEコマンドでは、文字として表示される

ファイルの内部データが表示される

左の内部データをアスキーコードとして、それに対応する文字に変換したもの。変換できなかったコードは「.」で示される

図 2.3　作成された文字ファイルの内容を DUMP プログラムで見る

DUMP プログラムは、任意のファイルの内容を、順に1バイトごとの16進表示でタイプアウトするものです。

ここで作成した文字ファイルの文字データは、コンピュータの内部やディスクの内部では、図2.3に表示されているように、1文字ごとにそれぞれの数値として表現され、取り扱われています。この数値がアスキーコードであり、上のリストから1文字が1バイトの数値で表されていることがわかります。

アメリカンスタンダードであるアスキーコードには、当然ながらカタカナなどは含まれていません。つまり、本来のアスキーコード表には文字が割り当てられていない空きエリア(80_H～FF_H)があり、その一部分($A0_H$～DF_H)にカタカナを割り当てたものが、JIS 規格「JIS-C 6220」として制定されている日本版アスキーコードです。現在の日本のパーソナルコンピュータのほとんどは、さらにその空きエリア(80_H～$9F_H$、$E0_H$～FF_H)に、各社独自のグラフィックキャラクタなどを割り当てています。このような観点から、巻末 APPENDIX の日本語版アスキーコード表をもう一度見直してみてください。

2.2 JIS 漢字コード

■ JIS 漢字コードの仕組み

JIS 漢字コードは JIS 規格「JIS-C 6226」に基づくものであり、1 文字を 2 バイト(16 ビット)で表します。つまりアスキーコードの 1 バイト系に対して、JIS 漢字コードは 2 バイト系であるわけです。

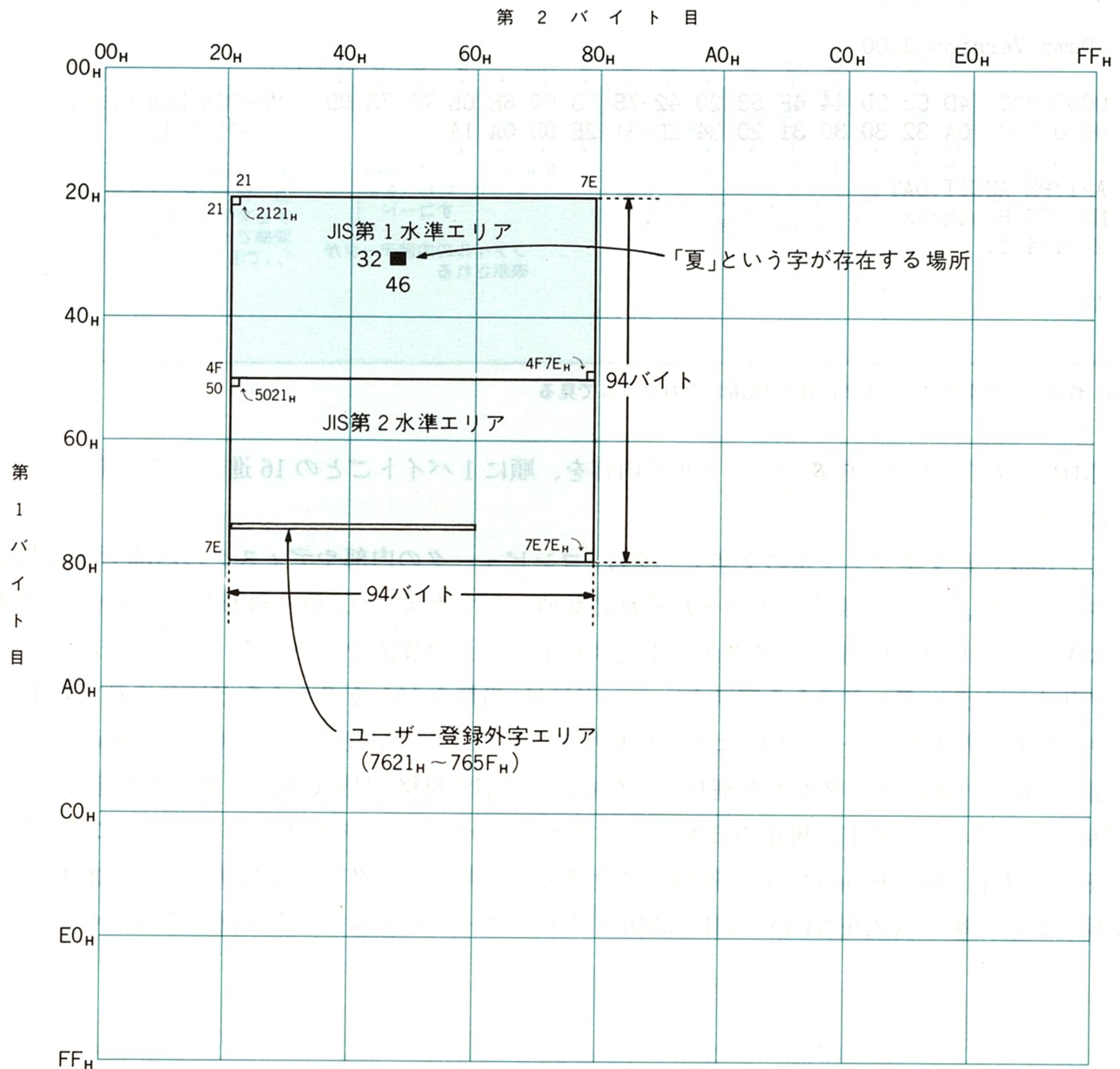



図 2.4 JIS 漢字コードに割り当てられているエリアマップ

2バイトで表せる数値の範囲は、16進数の0～FFFF$_H$(10進数の0～65535)であり、つまり約6万5千種類の文字を表すことが可能になります。JIS漢字コードは、前ページの**図2.4**に示すように、FF$_H$×FF$_H$の空間のなかで、2121$_H$および7E7E$_H$を対角線とするエリア(94×94＝8836文字)内に、記号、英数字、ひらがな、カタカナ、ギリシャ文字、ロシア文字などを含めた、JIS第1水準および第2水準の漢字を割り当てています。

このマップは、次節で解説するシフトJIS漢字コードで再登場し、この意味がさらにはっきりすることになります。

また、JIS漢字コードと実際の文字との対応を見るための一覧表——「JIS漢字コード表」の一部が、巻末のAPPENDIXにありますので参照してください。その表から、いくつかの漢字や記号などのコードを拾い出してみましょう。

愛＝3026$_H$　鳥＝3128$_H$　夏＝3246$_H$　海＝3324$_H$
〒＝2229$_H$　A＝2341$_H$　α＝2641$_H$　あ＝2422$_H$

さて、コンピュータ上で使われる日本語は、普通、本節で解説する2バイト系コードの文字と、前節の1バイト系コードの文字(つまり、全角文字と半角文字)とが混在しています。そのため、JIS漢字コードの場合は、その全角文字と半角文字との識別(切り替え)を、「KI(漢字IN)コード」、および「KO(漢字OUT)コード」と呼ぶ2つの制御コードによって行っています。この方式を、**KI／KO挿入方式**と呼んでいます(このことは、さらに次節でも解説します)。

この全角文字と半角文字との切り替え制御は、具体的には**ESCコード**(エスケープコード：1B$_H$)を使った「エスケープシーケンス」によって行われます。JIS漢字コードを使用している身近な例には、PC-9800シリーズのN$_{88}$-日本語BASICがありますが、その KI／KO コードの実際は次のようなものです。

KIコード …… 2バイト系の全角文字に切り替える合図 → [ESC] [K]

つまり、エスケープコードの1B$_H$、それに続けて4B$_H$(アスキー文字のK)の2バイトを、全角文字列の先頭に挿入する。

KOコード …… 1バイト系の半角文字に切り替える合図 → [ESC] [H]

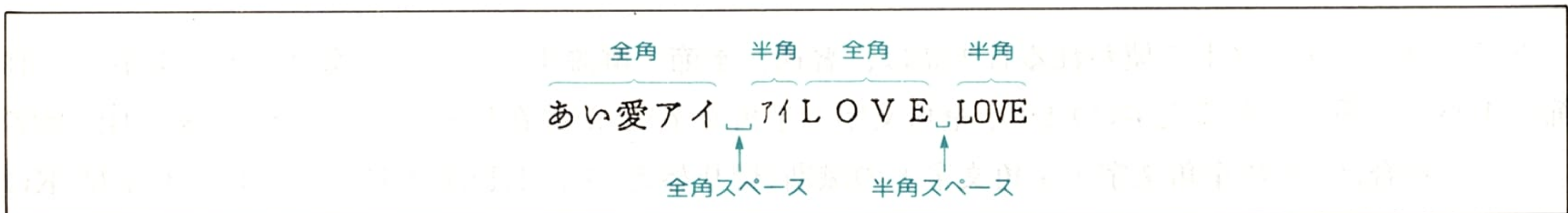
つまり、エスケープコードの1B$_H$、それに続けて48$_H$(アスキー文字のH)の2バイトを、半角文字列の先頭に挿入する。

実際の日本語文におけるこれらの具体例は、のちほど示します（KI／KOコードは、ほかのシステムでは別のコードが使われる場合もあります）。

ところが、全角／半角の切り替えのたびに、いちいちこのKI／KO制御コードを挿入することは能率も悪く、また、日本語処理上のいろいろな問題を含んでいます。そこでMS-DOSでは、このJIS漢字コードをもとに、それぞれの文字を、KI／KO制御コードを挿入しなくても、全角／半角の区別が可能になるように、JIS漢字コードをある規則をもとに巧妙にシフトした（割り当てた）、**シフトJIS漢字コード**を採用しています。このコードによって、従来の欠点が大幅に改善されましたが、それについては順次解説していきましょう。

■ N_{88}-日本語BASIC上の文字コード

さて次に、JIS漢字コードを使用しているシステムの1つであるN_{88}-日本語BASIC上で、簡単な日本語文を作成し、その実際の内部コードを見てみましょう。

```
   全角      半角  全角    半角
あい愛アイ　ｱｲＬＯＶＥ LOVE
          ↑            ↑
      全角スペース  半角スペース
```

このような2バイト系の全角文字と、1バイト系の半角文字の混在した文字列を、BASICプログラムのREM（リマーク）文を利用して作成します。BASIC上での日本語入力に関しては、BASIC言語のユーザーズマニュアルをご覧ください。全角文字は、もちろん日本語入力モードで入力しますが、半角文字は日本語入力モードからではなく、通常入力モードから入力してください。NECが提供する日本語入力システムから入力する半角文字は、標準的な入力モードでは、アスキーコードでもなくJIS

```
(N88-日本語BASICを起動する)

Disk version
How many files(0-15)? ↵
NEC N-88 BASIC(86) version 6.0
Copyright (C) 1983 by NEC Corporation / Microsoft Corp.
469988 Bytes free
Ok
     半    全      半    全     半
100 'あい愛アイ　ｱｲＬＯＶＥ LOVE ↵ ……行番号100の1行をリマーク文として全角・半角に注意して入力する
              └─全スペース └─半スペース
SAVE "BASKNJ",A ↵ ……ファイル「BASKNJ」としてアスキーセーブする

Ok
```

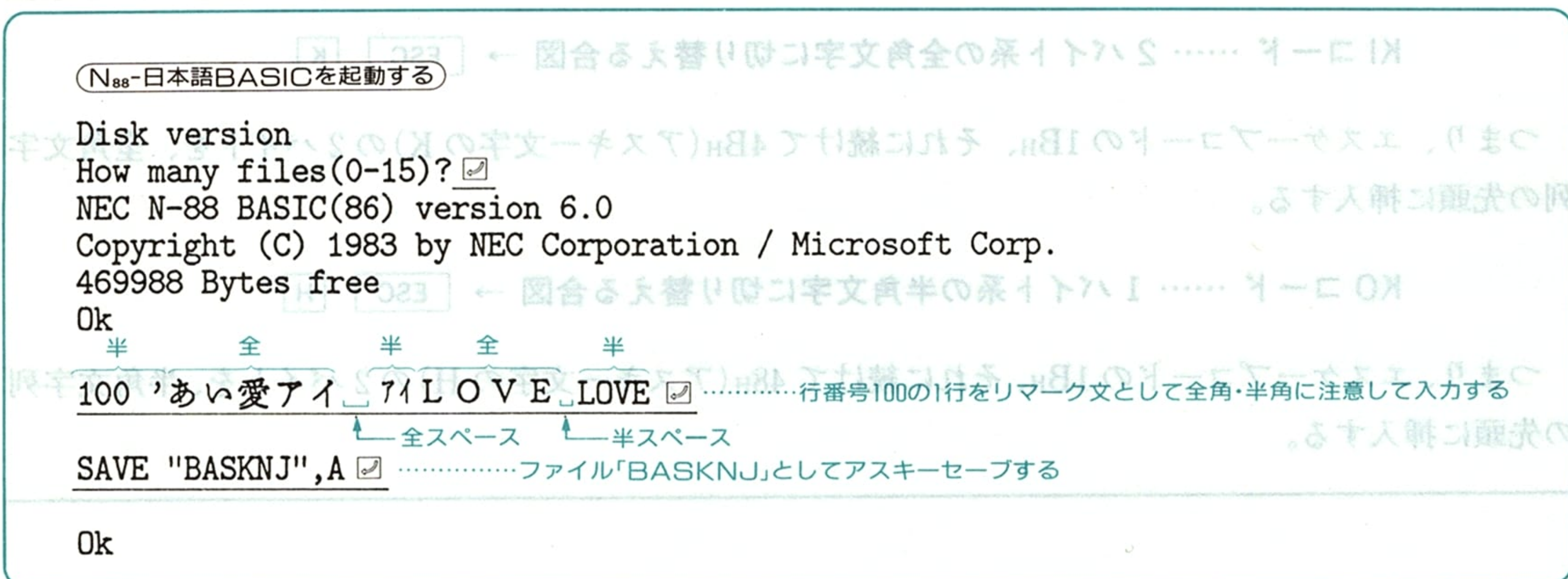



図2.5　N_{88}-日本語BASICによる日本語の文字ファイルの作成

漢字コードでもない、NEC 独自の特殊な 2 バイト系の半角文字コードとなり、一般性に欠けますので注意が必要です(1 バイト系の半角文字も、モード切り替えにより入力可能)。

このように行番号 100 のリマーク文として文字列を入力し、適当なファイル名 **BASKNJ** としてアスキーセーブによりディスクに格納します。この操作でディスク上には、行番号 100 の、たった 1 行のファイル **BASKNJ** が作成されました。

では、このファイルの内容がどのような文字コードでできているかを、ファイルダンププログラムで調べてみましょう。BASIC 上のダンププログラムが手元にあればすぐ調べられますが、ここでは 2.4 章で紹介する、「N88-BASIC ⇄ MS-DOS」のファイル変換プログラムを使って、BASIC ファイルの **BASKNJ** を、MS-DOS のディスク上に、**BASKNJNC**(ちなみに NC とは、Non-Convert の意味でつけた)というファイル名で移し替えた後、そのファイルの内容を MS-DOS の DUMP プログラムで調べてみます。

```
A>DUMP BASKNJNC⏎

 Dump Version 3.00

00000000  31 30 30 20 27 1B 4B 24-22 24 24 30 26 25 22 25   100 '.K$"$$0&%"%
00000010  24 21 21 1B 48 B1 B2 1B-4B 23 4C 23 4F 23 56 23   $!!.Hｱｲ.K#L#O#V#
00000020  45 1B 48 20 4C 4F 56 45-0D 0A 1A 00 00 00 00 00   E.H LOVE........

A>TYPE BASKNJNC⏎
100 '$"$$0&%"%$!!ｱｲ#L#O#V#E LOVE

A>
```


図 2.6 N88-日本語 BASIC による日本語の文字ファイルの文字コードを見る

このダンプ結果を、巻末 APPENDIX の日本版アスキーコード表、および、JIS/シフト JIS 漢字コード表と比較してみてください。1 バイト系のアスキーコード文字と、2 バイト系の JIS 漢字コード文字、その両者を区別するために挿入されている「KI/KO コード」の様子がよくわかります。このファイル(N88-日本語 BASIC により作成された日本語文ファイルを MS-DOS のファイル上に移したもの)は、MS-DOS 上のファイルに変換されてはいるものの、文字コードがシフト JIS 漢字コードではないため、シフト JIS 漢字コードを使っている MS-DOS 上では、TYPE コマンドでタイプアウトしても正しく表示されません。

2.3 シフトJIS漢字コード

前節で解説したJIS漢字コードは、2バイト系の文字と1バイト系の文字とを区別する手段として、「KI」および「KO」と呼ぶ「切り替え制御コード」(それぞれ2バイトからなる)を、文字列の全角/半角の境目ごとに挿入しなければなりません。したがって、この**KI/KOコード挿入方式**は、文字ファイル全体のデータ数(データ長)が、「文字データ」+「KI/KOコード」となるためにデータの効率が悪く、しかも、全角/半角文字の組み合わせによって、挿入されるKI/KOコードの数が変化するために、データ長が不定であり、データ処理上のいろいろな問題が発生します。

その一例を示しましょう。MS-DOSには、個々のディスクに**ボリュームラベル**というディスクの識別名を付けることができ、DIRコマンドなどではそのディスク名が表示されます。ボリュームラベルを登録するには、スイッチ「/V」付きのFORMATコマンドを実行するか、あるいはLABELコマンド(MS-DOSバージョン3.*x*以降に付属のコマンド)を実行します。そのFORMATコマンドの実行画面を次に示します。

```
A>FORMAT B:/V⏎ ……………FORMATプログラムによりボリュームラベルを登録する
Format Version 4.10

    ⋮ (途中省略)

ボリュームラベルを入力してください
漢字<全角>は５文字、英数字<半角>は１１文字まで、必要なければリターンキー
                           └─┴─半角
ボリュームラベル＝歴史A1現代2⏎ ……………ボリュームラベルを入力する

別のディスクをフォーマットしますか(Y/N) N⏎
A>DIR B:⏎ ……………DIRコマンドでもボリュームラベルが表示される

 ドライブ B: のディスクのボリュームラベルは 歴史A1現代2
 ディレクトリは B:¥

ファイルが見つかりません.

A>
```

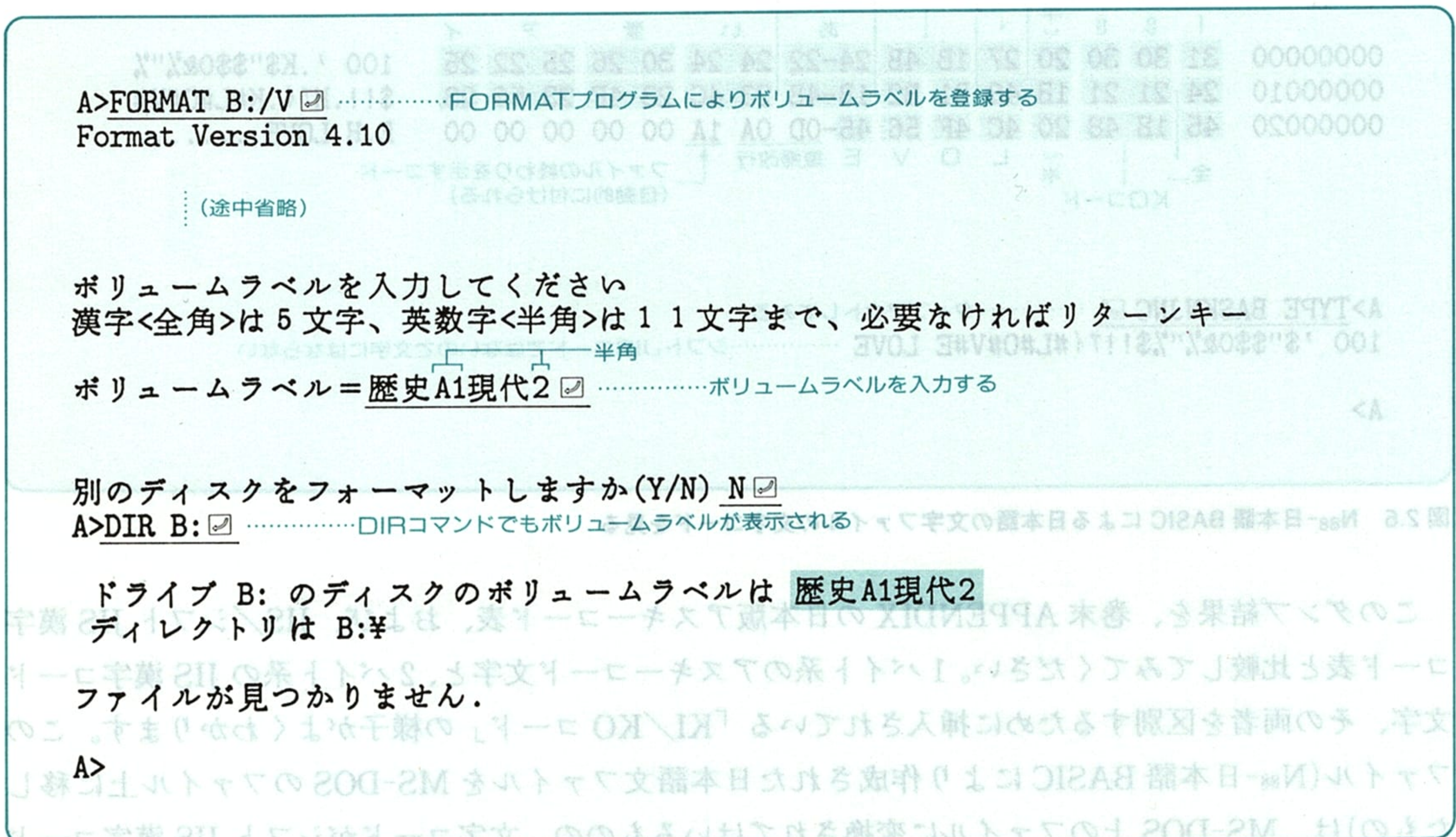
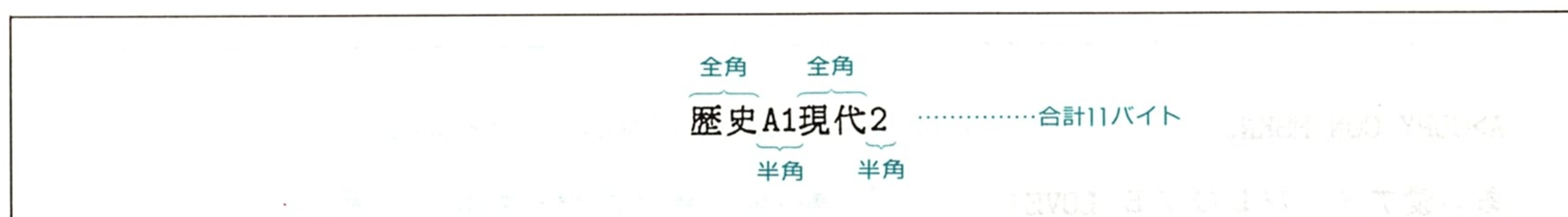



図2.7 FORMATコマンドによるボリュームラベルの登録

ここで、「漢字－5 文字、英数字－11 文字」と表示されているのは、ディスク名を表す文字データを収容するエリアが、11 バイト分用意されているからです。漢字は 1 文字が 2 バイトで表されるため、11／2＝5.5 となり、漢字のみを使った場合は、5 文字以下でなければなりません(半角文字がもう 1 文字使えますが)。

MS-DOS では、2 バイト系の文字と 1 バイト系の文字との間に、KI／KO コードのような制御コードが必要でないため、全角／半角の自由な組み合せが可能です。つまり、全角／半角をどのように組み合せた文字列であっても、合計のバイト数が 11 バイト以下であればよいわけです。たとえば、上の実行例に示したように、図書管理データが格納されているディスクであれば、

全角 全角 歴史A1現代2 ……………合計11バイト 半角 半角

というディスク名を付けてもよいわけです(全角文字が 4 文字、半角文字が 3 文字で合計 11 バイト)。

このような、全角／半角の混在が自由にできるのは、MS-DOS が、KI／KO などの制御コードがいっさい不要な**シフト JIS 漢字コード**を採用していることによります。もしこれが、KI／KO コードを挿入しなければならない **JIS 漢字コード**であれば、どのようなことになるでしょう。もっとも極端な例は、ディスク名が次のように 1 文字ずつ、全角・半角・全角と並んだ場合です。

```
   全  半  全  半  全  半  全  ←文字データの合計 11 バイト(2×4＋1×3)
 ↑  ↑  ↑  ↑  ↑  ↑  ↑
 KI  KO  KI  KO  KI  KO  KI   ←制御コードの合計 14 バイト(2×7)
```

このディスク名に必要な全バイト数は、文字そのものが 11 バイト、KI／KO コードが 14 バイトで、計 25 バイトになります。つまり、シフト JIS 漢字コードを採用した場合には、11 バイトのエリアを用意しておけばよかったものが、JIS 漢字コードを採用した場合は、25 バイト分のエリアを用意しておかなくてはならないわけです(エリアの全部が使われても使われなくても)。

このようなことから、JIS 漢字コードを採用しているシステムにおいては、全角文字／半角文字を自由に混在させることが困難であり、現実的には、入力項目によって、あらかじめ使用文字の種類を全角／半角のいずれかに指定して、混在を許さないものが多いのです。

しかし、このような制限は、ユーザーにきわめて不自由を強いるものであり、日本語を扱う仕事では致命的な欠点となります。そこで、これらの問題点の元凶である、2 バイト系／1 バイト系の切り替えコードを使用しない「シフト JIS 漢字コード」がマイクロソフト社等によって考案され、MS-DOS や OS/2、CP/M などの日本語処理システムに採用されたのです。

■ MS-DOS 上の文字コードの実際

ではMS-DOS上で、簡単な日本語文を作成し、シフトJIS漢字コードの実際を見てみましょう。前節でも例にした「あい愛アイ　ｱｲＬＯＶＥ love」という内容の文字ファイルを、COPYコマンドを使って作成してみましょう(半角／全角に注意してください)。使い慣れているエディタがあれば、それを使ってください。前節でも述べましたが、NECの日本語入力システムから入力する半角文字は、設定によってはアスキーコードでもなく、JIS漢字コードでもない、NEC独自の2バイト系の半角文字コードとなり一般性がありませんので、半角文字は通常入力モードで入力してください(ATOKや松茸、VJEなどでは、そのようなことはありません)。

```
A>COPY CON MSKNJ⏎ ……………COPYコマンドを使って、ファイル「MSKNJ」を作成する
あい愛アイ　ｱｲＬＯＶＥ LOVE⏎ ……………全角・半角に注意して入力する(前回のものと同じ内容)
^Z⏎ ……………CTRL+Z⏎
        1個のファイルをコピーしました.
A>
```

図 2.8　MS-DOS 上で簡単な文字ファイルを作成する

この作業により、ディスク上には、半角／全角文字が混在した文字列のファイル、MSKNJが作成されました。ではこのファイルの内容が、どのような文字コードでできているのかを、MS-DOSシステムディスクに標準装備されているDUMPプログラム「DUMP.EXE(COM)」で調べてみましょう。

```
A>DUMP MSKNJ⏎ ……………COPYコマンドで作成したファイルの内部データをダンプする

Dump Version 3.00
            あ    い    愛    ア    イ    ␣     ア イ Ｌ
00000000  82 A0 82 A2 88 A4 83 41-83 43 81 40 B1 B2 82 6B   あい愛アイ　ｱｲＬ
00000010  82 6E 82 75 82 64 20 4C-4F 56 45 0D 0A 1A         ＯＶＥ LOVE...
            Ｏ    Ｖ    Ｅ    ␣ L  O  V  E  復帰改行コード
A>TYPE MSKNJ⏎
あい愛アイ␣ｱｲＬＯＶＥ␣LOVE ……………シフトJISコードなので文字として表示される

A>
```

図 2.9　作成した MS-DOS 上の文字ファイルの文字コードを調べる

このファイルのダンプ結果を、APPENDIX の日本版アスキーコード表、JIS／シフト JIS 漢字コード表と比較してみてください。1バイト系のアスキーコード文字と、2バイト系のシフト JIS 漢字コード文字のデータのみが存在し、それらを区別する KI／KO コードなどは存在していないことがよくわかります。なぜシフト JIS 漢字コードには、KI／KO コードなどが不要なのでしょう。そのことについてはこのあと解説します。

■ MS-DOS 版 N_{88}-日本語 BASIC 上の文字コード

NEC では、スタンドアローン(マシンに組み込み)の N_{88}-日本語 BASIC を、MS-DOS 上に移植した、N_{88}-日本語 BASIC86(MS-DOS 版)と、そのコンパイラを提供しています。前節で確認した 94 ページの図 2.5 のように、スタンドアローンの N_{88}-日本語 BASIC では、日本語が JIS 漢字コードで取り扱われますので、なにかと不自由な思いをされている方も多いのではないかと思います。ところが MS-DOS 版では、これがシフト JIS 漢字コードに変更されています。それを確認してみましょう。

前節と同じように、行番号 100 のみの文字ファイルを作成した後、DUMP プログラムでその内部コードを確認してみます。その実行例を次に示します。

```
A>N88BASIC ⏎ ……………MS-DOS上のN88-日本語BASICを起動する

NEC N-88 BASIC(86) version 6.0
Copyright (C) 1984,88 by NEC Corporation
322940 Bytes free
Ok
100 'あい愛アイ ｱｲL O V E LOVE ⏎ ……………全角・半角に注意してリマーク文として入力する
SAVE "MSBASKNJ",A ⏎ ……………ファイル「MSBASKNJ」としてアスキーセーブする
Ok
SYSTEM ⏎ ……………BASICを終了してMS-DOSにもどるコマンド

A>
```

図 2.10 MS-DOS 版 N_{88}-日本語 BASIC による文字ファイルの作成

この操作で、行番号 100 のリマーク文が、MSBASKNJ.BAS としてアスキーセーブされました。ファイル名のファイルタイプ(拡張子)の「.BAS」は、この部分の指定を省略した場合に自動的に付けられるものです。ではこれをダンプしてみましょう。

```
A>DUMP MSBASKNJ.BAS ⏎ ……………MS-DOS上のBASICで作成したファイルの内部データをダンプする

 Dump Version 3.00        ┌→先にCOPYコマンドで作成したものと同一（図2.9参照）
           1  0  0  ␣  '  あ     い     愛     ア     イ
00000000  31 30 30 20 27 82 A0 82-A2 88 A4 83 41 83 43 81    100 'あい愛アイ
00000010  40 B1 B2 82 6B 82 6E 82-75 82 64 20 4C 4F 56 45     ｱｲＬＯＶＥ LOVE
00000020  0D 0A 1A                                            ...
          復帰改行 └──ファイルの終わりを示すコード
A>TYPE MSBASKNJ.BAS ⏎
100 'あい愛アイ␣ ｱｲＬＯＶＥ␣LOVE …………シフトJISコードなので文字として表示される

A>
```

図 2.11　MS-DOS 版 N_{88}-日本語 BASIC 上の文字コードの確認

このようにMS-DOS版 N_{88}-日本語BASICで扱う文字コードは、シフトJIS漢字コードであることが確認されます。

■ シフトJIS漢字コードの仕組み

ではここで、シフトJIS漢字コードの原理を、JIS漢字コードと比較して解説しましょう。JIS漢字コードには、KI／KOコードの挿入が必要であることの理由、また、シフトJIS漢字コードには、その必要がない理由が明らかになるでしょう。

次ページの図2.12は、前節の図2.4で示したJIS漢字コードのマップに、シフトJIS漢字コードのマップを重ねたものです。前節でも解説したように、2バイト系の文字は、1つの文字を2バイトのコード（第1バイト目および第2バイト目）で表しますので、このマップは $FF_H \times FF_H$ の空間、つまり256×256の文字エリアをもつことができます。このエリアのなかで、JIS漢字コードは、2121_H および $7E7E_H$ の2点を対角線とするエリア（94×94文字）に割り当てられています。

左端の縦軸は、1バイト系文字の日本版アスキーコード（JIS-C6220：以降「アスキーコード」と記す）のマップです。1バイト系は横軸の広がりがありませんので、縦軸のみで示されています。このアスキーコードのマップと、JIS漢字コードのマップとを対比してみてください。

まずJIS漢字コードは、1バイト目も2バイト目も 21_H〜$7E_H$ の値であり、アスキーコードで使われている範囲のコードであることがわかります。つまり、1バイト系コードと、2バイト系コードが連続して混在している場合、両者を分離する手掛りはまったくなく、その区別は不可能であることになります。したがって、両者を区別するためには、その境界に合図となる特殊なコードを、あらかじめ挿入しておくほかに手はないのです。これが前節でその実際を見て解説した、KI／KOコードなのです。

第 2 バイト目

00H 20H 40H 60H 80H A0H C0H E0H FFH

アスキーコード
コントロール・キャラクタ
記号／数字／英大文字
英小文字／記号
未定義
記号／カタカナ
未定義

第1バイト目

94バイト
JIS第1水準エリア
JIS第2水準エリア
31バイト
8140H
シフトJIS第1水準エリア
第2水準エリア 9F7EH
8180H
シフトJIS第1水準エリア
第2水準エリア 9FFCH
98A0H 以降第2水準
94×2バイト
16バイト
E040H
シフトJIS第2水準エリア EF7EH
E080H
シフトJIS第2水準エリア EFFCH
将来の拡張用
将来の拡張用

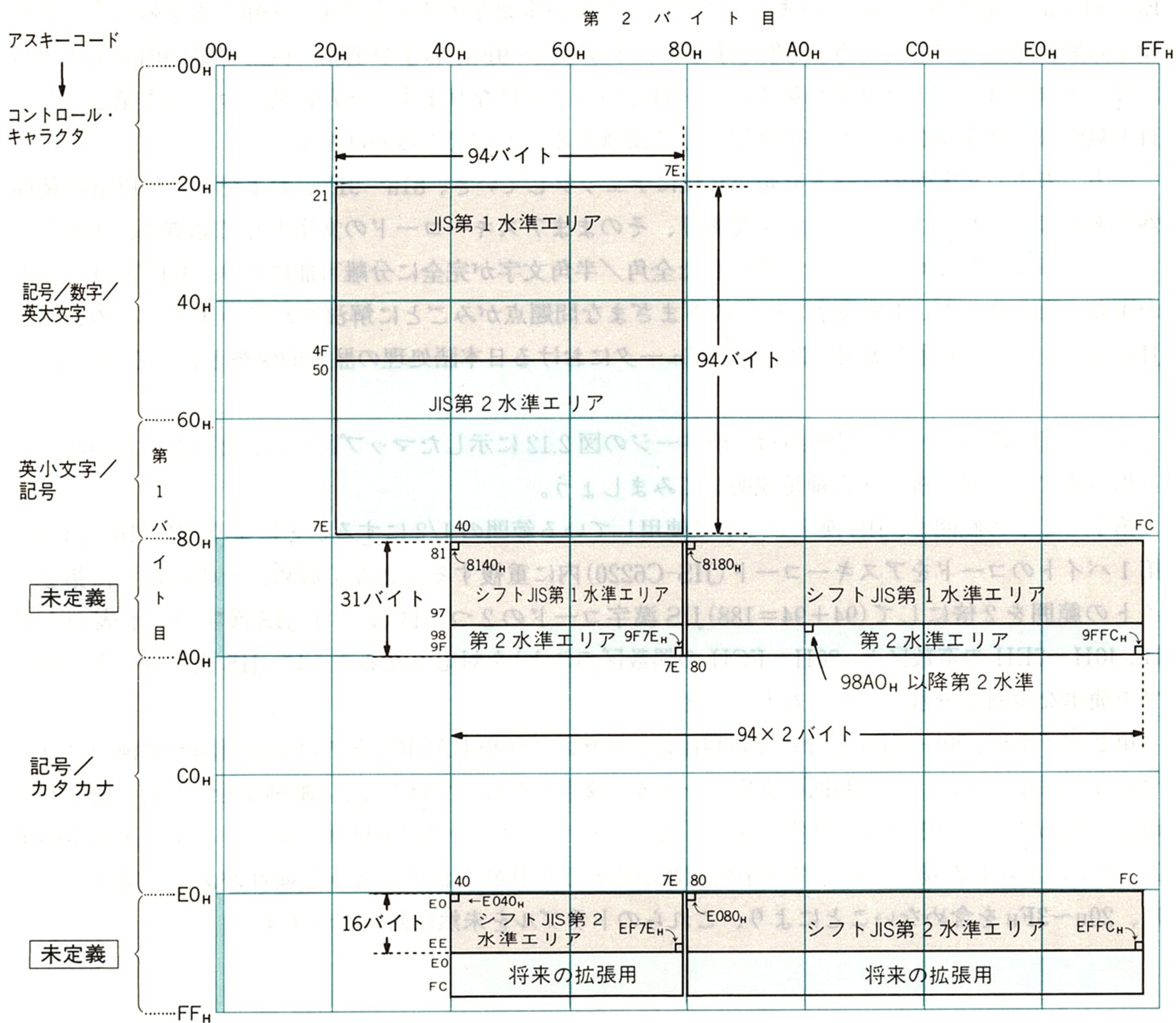



図 2.12　JIS／シフト漢字コードの割り当てマップ

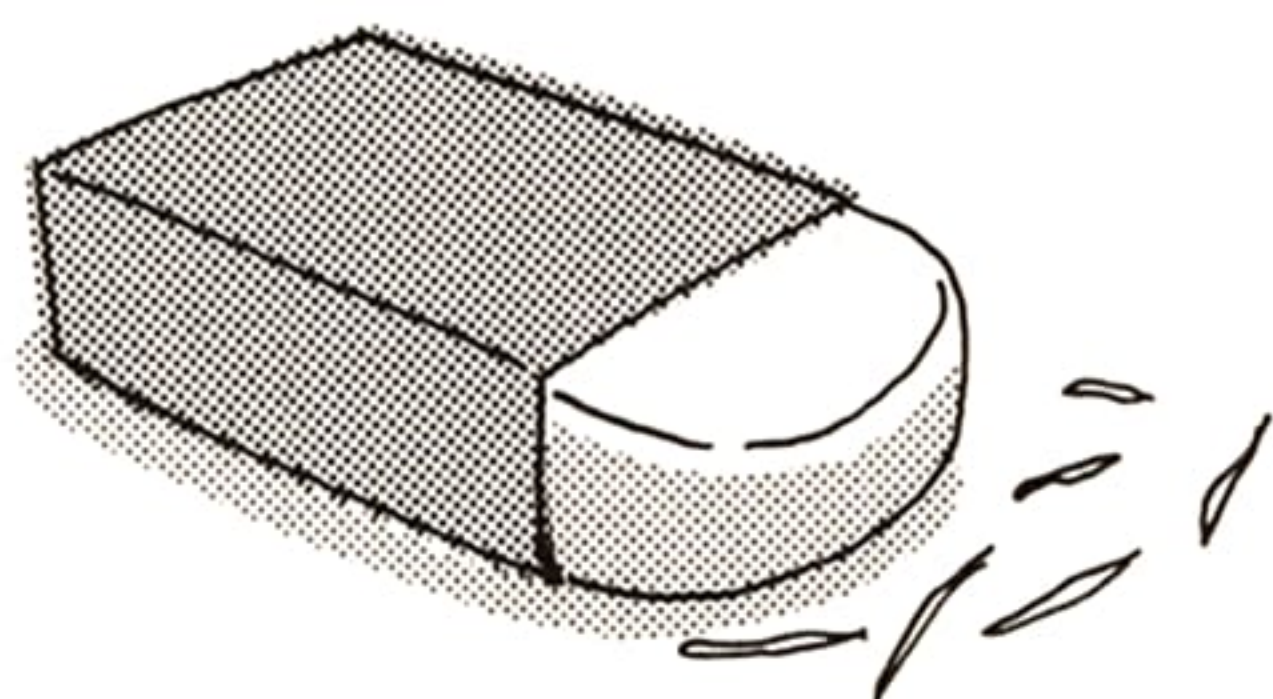
では次に、アスキーコードのマップと、シフトJIS漢字コードのマップとを対比してください。シフトJIS漢字コードの第1バイト目のコードは、アスキーコードの未定義の部分(81_H～$9F_H$、および$E0_H$～FC_H)に割り当てられています。ここのところが重要なポイントです。未知の文字列のコードを頭から順にチェックしていき、対象とするコードが81_H～$9F_H$、および$E0_H$～FC_Hの範囲内のものであれば、それは2バイト系文字の第1バイト目ということになります。それに続くコードを第2バイト目と見れば、漢字やひらがな等の文字として認識することができるわけです。

一方、未知の文字列のコードを頭から順にチェックしていき、81_H～$9F_H$、および$E0_H$～FC_Hの範囲外であれば、それは1バイト系文字であり、そのままアスキーコードの文字として認識されます。

このような単純なロジックで、混在した全角／半角文字が完全に分離可能になり、KI／KOコードが不要となりました。JIS漢字コードのさまざまな問題点がみごとに解決されています。このシフトJIS漢字コードの考案と実用化は、コンピュータにおける日本語処理の歴史的快挙と言えるでしょう。

シフトJIS漢字コードの仕組みは、前ページの図2.12に示したマップ図から、だいたいおわかりかと思いますが、別の視点から補足説明してみましょう。

第1バイトの範囲を、JIS漢字コードで使用している範囲の1/2にすることにより(31＋16＝94/2)、第1バイトのコードをアスキーコード(JIS-C6220)内に重複することなく収め、その代わりに第2バイトの範囲を2倍にして(94＋94＝188)JIS漢字コードの2つの区(シフトJIS漢字コード表の上では、40H～7EHの奇数区と、80H～FCHの偶数区のこと)を対応させるように、JIS漢字コードのマップを簡単な規則で分割シフトしたものです。

第2バイトに、20_H～$3F_H$を含めない理由は、アセンブラやBASIC、そのほかの言語や各種のプログラムなどによっては、この範囲の文字コードを「文字」としてではなく、「制御記号」として使うものがあるためです。たとえば、文字列をくくる「'」や「"」のコードと同じものが、2バイト系文字の第2バイトに出現した場合、そこで文字列定数が終わりと判断されてしまう危険性があり、第2バイトに、20_H～$3F_H$を含めないことにより、これらのトラブルを未然に防いでいます。

2.4 JIS漢字コードとシフトJIS漢字コードの相互変換

これまでの解説で、JIS 漢字コード、およびシフト JIS 漢字コードの成り立ちや、その違いなどについて、ある程度理解できたことと思います。シフト JIS 漢字コードは、割合に簡単な規則にしたがって JIS 漢字コードを変換したものですので、プログラマが、それら相互の変換プログラムを作成することは、さほどむずかしいことではありません。C 言語を使えば、ほんの短いソースプログラムで済むでしょう。

さて、パーソナルコンピュータが実務に本格的に利用されるようになり、ほかのコンピュータシステムとのデータ交換が必要な状況になってきました。そこで本節では、JIS 漢字コードを使用しているシステムで作成されたディスク上のデータファイルを、MS-DOS 上のシフト JIS 漢字コードのデータファイルに変換する実例を示しましょう。

このような変換ソフトは、何種類か市販されていますが、ここでは、PC-9800 シリーズの MS-DOS のシステムディスク(バージョン 3.x 以降)に付属している、N88/MS-DOS ファイルコンバータ **FILECONV.EXE** による実行例を紹介します。ただしこのプログラムは、本来、文字コードを変換するだけでなく、スタンドアローンの N88-BASIC のディスクファイルと、MS-DOS のディスクファイルとのファイル交換(変換)プログラムであり、双方向のファイル変換が行えます。

当然ながら、互いに相手のシステムで使っているディスクの読み/書きは、そのままでは不可能ですが(その逆も同じ)、このソフトウェアによって、相手のディスク上の任意のファイルを、自分のディスク上にもってくることができ、互いのファイルの読み/書きが(つまり利用が)可能になります。

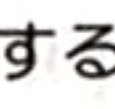
さて、ファイルそのものの変換はできたとしても、N88-日本語 BASIC における文字コードは JIS 漢字コードであり、MS-DOS ではシフト JIS 漢字コードです。したがって、ファイル交換をしても、互いの文字コードが異なるため、そのなかの文字データを「文字」として取り扱うことができません。そこでこのファイルコンバータは、ファイル転送を行う際に、ファイル内部の文字コードの変換(JIS 漢字コードとシフト JIS 漢字コードとの変換)も同時に行うようになっています(変換する/しないの選択可能)。

次項では、「JIS 漢字コード⇄シフト JIS 漢字コード」の変換に注目して、N88/MS-DOS ファイルコンバータの実行例を示しましょう。

■ N88/MS-DOS ファイルコンバータによる変換

NEC では、スタンドアローンの N88-日本語 BASIC を、MS-DOS 上に移植した N88-日本語 BASIC86 と、そのコンパイラを提供しています。これらの間では、「スタンドアローン版 BASIC ⇄ MS-DOS 版 BASIC」の各種のファイル交換も必要になるでしょう。そのため、さきに述べたように、「N88-BASIC ⇄ MS-DOS」のファイル変換プログラムが、MS-DOS バージョン 3.x 以降のシステム

ディスクに FILECONV.EXE というファイル名で付属しています。このディスクファイル変換ソフトは、ファイル転送を行うと同時に、指定により「JIS 漢字コード⇄シフト JIS 漢字コード」の文字コード形式の変換も行えるようになっています。

では、この N88／MS-DOS ファイルコンバータ「FILECONV.EXE」を、作業用の MS-DOS システムディスクにコピーしておき、「JIS 漢字コード→シフト JIS 漢字コード」のコード変換の実行例を示しましょう。この逆の変換も可能ですが、操作はほとんど同じですので省略します。

まず、ファイル変換の対象ファイルとして、N88-日本語 BASIC 上で、適当な日本語の文字ファイルを作成しておきます。ここでは 2.2 章で作成した BASIC ファイルの BASKNJ を利用しましょう。94 ページの図 2.5 の作成例をもう一度参照してください。また、このファイルの文字コードは、95 ページの図 2.6 のダンプリストから、JIS 漢字コードであることを再度確認しておいてください。

ドライブ A：には、プログラムファイル「FILECONV.EXE」をコピーした MS-DOS のシステムディスク、ドライブ B：には、文字ファイル「BASKNJ」を含んだ N88-日本語 BASIC のディスクをセットして、この変換プログラムを起動します。起動は次のコマンドで行います。

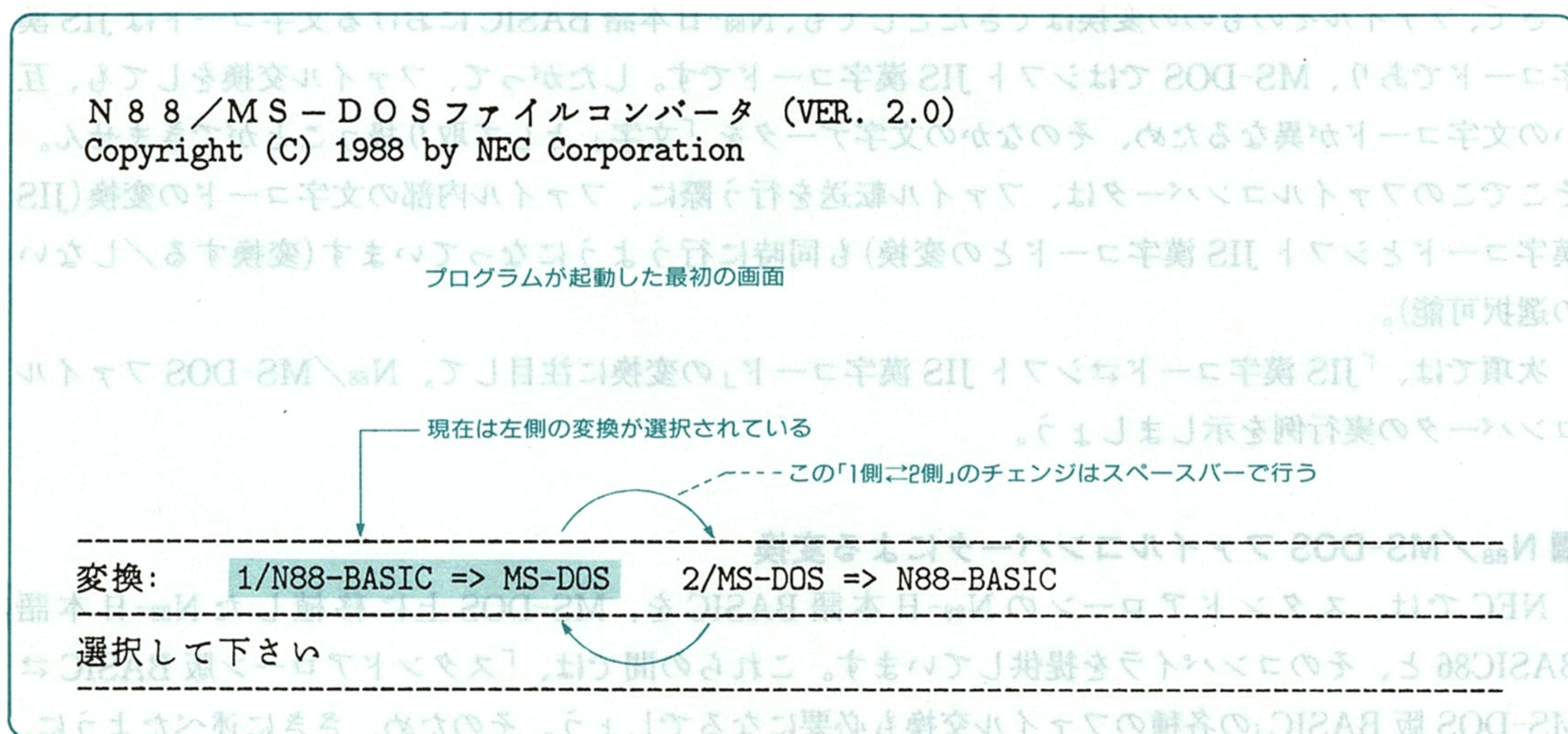
```
A>FILECONV ↵
```

プログラムが起動すると、次の画面が表示されます。この画面は第 1 画面で、メディア変換の方向——N88-BASIC ディスク⇄MS-DOS ディスクの「→」／「←」——の選択をするものです。このプログラムは、すべての画面がコマンドメニューによる対話形式になっていますので、メニューにしたがって容易に実行することができます。

```
Ｎ８８／ＭＳ－ＤＯＳファイルコンバータ (VER. 2.0)
Copyright (C) 1988 by NEC Corporation

------------------------------------------------------------------
変換:  1/N88-BASIC => MS-DOS    2/MS-DOS => N88-BASIC
------------------------------------------------------------------
選択して下さい
------------------------------------------------------------------
```

プログラムが起動した最初の画面

現在は左側の変換が選択されている

この「1側⇄2側」のチェンジはスペースバーで行う

変換は「1側」でよいので ↵ を入力する

```
----------------[ N88-BASIC => MS-DOS ]----------------
-----------------------------------------------------------
変換方法は   1/FILE   2/VOLUME
-----------------------------------------------------------
選択して下さい
-----------------------------------------------------------
```

指定したファイルの変換を行う
スペースバーによりチェンジできる
ディスク上の全ファイルを変換する場合に選択する

ファイル単位の変換を行うので「1側」を選択して⏎する

```
-----------------------------------------------------------
N88-BASICのドライブは  B:⏎
-----------------------------------------------------------
```

BASICのディスクはドライブB：にセットしたので「B：⏎」と答える

```
-----------------------------------------------------------
MS-DOSのドライブは  A:⏎
-----------------------------------------------------------
```

MS-DOSのディスク(変換されたファイルがセーブされるディスク)はドライブA：なので「A：⏎」と答える

```
----------------[ N88-BASIC Directory ]----------------
menu    .       format  .nip    backup  .n88    setinf  .n88    xfiles  .n88
sysgen  .nip    format  .hd     recov   .hd     dir     .hd     backup  .hd
mkfont  .n88    switch  .n88    dicmen  .n88    DDconv  .n88    mouse   *cod
tele    .n88    XMODEM  *BIN    setup   .n88    BASKNJ
-----------------------------------------------------------
N88-BASICのファイル名は  BASKNJ
-----------------------------------------------------------
選択して下さい
-----------------------------------------------------------
```

連動している

上のディレクトリ表示のなかから、変換するファイル名をカーソル移動キー↑↓→←で選択する

カーソル移動キーでファイル名を選択後⏎する

```
-----------------[ N88-BASIC => MS-DOS ]-----------------
BASKNJ      =>
```

変換されるBASICのファイル名が表示されている

```
---------------------------------------------------------
MS-DOSのファイル名は  BASKNJC⏎
---------------------------------------------------------
```

……変換後のファイルに付けるMS-DOS上の適当なファイル名を入力する。⏎のみ入力すると、BASICのファイルと同名となる

⬇ 変換後のファイル名を入力して⏎する

```
-----------------[ N88-BASIC => MS-DOS ]-----------------
BASKNJ      =>  BASKNJC
```

……もとのファイル名、変換後のファイル名が示されている

スペースバーでチェンジできる(以降同様)

今回は通常のファイルなのでこちら側を選択する

```
---------------------------------------------------------
ランダムデータファイルですか     1/YES    2/NO
---------------------------------------------------------
選択して下さい
---------------------------------------------------------
```

⬇ 「2側」を選択して⏎する

「シフトJIS⇄JIS」の変換を行うかどうかの選択。ここでは行うので「1側」を選択

```
---------------------------------------------------------
日本語JISコードの変換は     1/YES    2/NO
---------------------------------------------------------
```

⬇ 「1側」を選択して⏎する

```
-----------------[ N88-BASIC => MS-DOS ]-----------------
BASKNJ      =>  BASKNJC
JISコード変換
KI/KOコード
```

「JIS→シフトJIS」へ変換され、不要になったKI/KOコードを、削除するか、ダミーコード(なんの働きもしないコード)にするかの選択。ダミーコードにした場合は、ファイルの長さが変わらない。ここでは削除する

```
---------------------------------------------------------
KI/KOコードは    1/DELETE    2/0FFFFH
---------------------------------------------------------
選択して下さい
---------------------------------------------------------
```

削除する　ダミーコードに置き換える

⬇ 「1側」を選択して⏎する

```
-------------------[  N88-BASIC  =>  MS-DOS  ]-------------------
BASKNJ      =>  BASKNJC
ＪＩＳコード変換
ＫＩ／ＫＯコード   DELETE
--------------------------------------------------------------------
パラメータを確認して下さい  1/YES   2/NO
--------------------------------------------------------------------
```

今までに選択した項目が表示されているので確認する

もし指定ミスがあれば、こちらを選択し⏎することにより再指定が可能

確認がOKならば⏎する

```
--------------------------------------------------------------------
変換は終了しました。別のファイルを変換しますか  1/YES   2/NO
--------------------------------------------------------------------
```

「1側」を選択して⏎する

```
-------------------[  MS-DOS Directory  ]-------------------
COPY2    COM   COPYA    COM   DICM     COM   EXE2BIN  EXE   FILECONV EXE
FIND     EXE   HDUTL    EXE   JOIN     EXE   LABEL    EXE   LIB      EXE
LINK     EXE   MAKE     EXE   MAPSYM   EXE   MORE     COM   PRINT    EXE
RECOVER  EXE   SHARE    EXE   SORT     EXE   SUBST    EXE   NECDIC   DRV
NECDIC   SYS   CONFIG   1     CONFIG   2     CONFIG   3     BASKNJC  BAS
--------------------------------------------------------------------
変換は終了しました。
--------------------------------------------------------------------
どれかキーを押して下さい
--------------------------------------------------------------------
```

MS-DOS側ディスクのディレクトリが表示される

「JIS→シフトJIS」に漢字コードが変換され、MS-DOSのディスク上に転送されたファイル。「BAS」はファイルタイプを付けなかったため、自動的に付加されたもの

適当なキーの入力によりMS-DOSにもどる

図 2.13　N_{88}／MS-DOS ファイルコンバータによる文字コード変換例

以上の操作で、N_{88}-日本語 BASIC 上のファイルが MS-DOS ファイルに変換され、MS-DOS のディスク上に転送されました。転送されたファイル BASKNJC.BAS をダンプして確認してみましょう(「.BAS」は、ファイルタイプの指定を省略したため、自動的に付けられたもの)。

```
A>DIR BASKNJC.*⏎ ……………MS-DOSのディスク上に作成されたファイルの確認

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

BASKNJC  BAS        35  89-08-11   19:58 ……………転送されたファイルが作成されている
        1 個のファイルがあります.
  117760 バイトが使用可能です.

A>DUMP BASKNJC.BAS⏎ ……………ファイルの内部データをダンプして確認する

 Dump Version 3.00   MS-DOS版N88-日本語BASICのもの(図2.11)と、まったく一致しており、
                     「JIS→シフトJIS」のコード変換が行われている

00000000  31 30 30 20 27 82 A0 82-A2 88 A4 83 41 83 43 81   100 'あい愛アイ
00000010  40 B1 B2 82 6B 82 6E 82-75 82 64 20 4C 4F 56 45    ｱｲＬＯＶＥ LOVE
00000020  0D 0A 1A                                            ...

A>TYPE BASKNJC.BAS⏎
100 'あい愛アイ␣ｱｲＬＯＶＥ␣LOVE ……………シフトJISコードなので文字として表示される

A>
```

図 2.14　シフト JIS 漢字コードに変換された MS-DOS 上の文字ファイルの確認

このダンプ結果から、N88-日本語 BASIC 上のファイルが MS-DOS ファイルに変換されたと同時に、図 2.4 の JIS 漢字コードが、シフト JIS 漢字コードに変換されていることが確認されます。

3章
RAMディスク、ディスクキャッシュ、EMS

「RAMディスク」「ディスクキャッシュ」「EMS」――この3つは、いずれもコンピュータ内に組み込んだ大容量のRAMボードなどによって大きなメモリ空間を作り出し、そのメモリをそれぞれのソフトウェアの操作によって、それぞれの機能を実現するという共通点があります。これらは「拡張メモリ／増設メモリ」に関する1つのテーマに包括されるといってもよいでしょう。

コンピュータには、「もっと速く」（処理速度）、「もっと広く」（メモリ空間）と、際限なく要求され続ける宿命があります。コンピュータの処理速度は、当然のことながらハードウェアにもソフトウェアにも総合的に関係します。しかしすでに持っているマシンのCPUの処理速度を上げるわけにはいきません。誰でも気軽にできるハードウェア的なことは、唯一、ディスク装置の高速化です。そこでハードディスクやRAMディスクなどの高速ディスク装置や、ディスクのアクセスを高速化するディスクキャッシュと呼ばれる仕掛けなどが広く使われるようになりました。

また、一般的なMS-DOSマシンにおいて、ユーザーが使えるメインメモリの容量は、最大640Kバイトですが、この制限は、プログラムや取り扱うデータが大きくなってきたため、最近のビジネスソフトなどの実行に支障をきたすようになってきました。その問題を解決するために登場したのが、「EMS」と呼ばれる、メモリを拡張するための規格です。

そこで本章では、ディスク装置を高速化するためのMS-DOSの設定や、RAMディスク、ディスクキャッシュなどについて解説し、さらにメモリを有効に使うことによって、大きなアプリケーションプログラムを実行したり、大きなデータを取り扱うことを可能にするEMSについて解説しましょう（なお、ハードディスクについては、別項で詳しく解説します）。

「RAM ディスク」、「ディスクキャッシュ」、「EMS」――この 3 つは、いずれもコンピュータ内に組み込んだ大容量の RAM ボードなどによって大きなメモリ空間を作り出し、そのメモリをそれぞれ巧妙なソフトウェアの操作によって、それぞれの機能を実現するという共通点があります。これらは「拡張メモリ／増設メモリ」に関する 1 つのテーマに包括されると言ってもよいでしょう。

コンピュータには、「もっと速く」(処理速度)、「もっと広く」(メモリ空間)と、際限なく要求され続ける宿命があります。コンピュータの処理速度は、当然のことながらハードウェアにもソフトウェアにも総合的に関係します。しかし、すでに使っているマシンの CPU の処理速度を上げるわけにはいきませんので、私たちにできるハードウェア的なことは、唯一、ディスク装置の高速化です。そこでハードディスクや RAM ディスクなどの高速ディスク装置や、ディスクのアクセスを高速化するディスクキャッシュと呼ばれる仕掛けなどが広く使われるようになりました。

また、一般的な MS-DOS マシンにおいて、ユーザーが使えるメインメモリの容量は、最大 640K バイトですが、この制限は、プログラムや取り扱うデータ量が大きくなる一方の、最近のビジネスソフトなどの実行に支障をきたすようになってきました。その問題を解決するために登場したのが、「EMS」と呼ばれるメモリを拡張するための規格です。

そこで本章では、ディスク装置を高速化するための MS-DOS の設定や、RAM ディスク、ディスクキャッシュなどについて解説し、さらにメモリを拡張することによって、大きなアプリケーションプログラムを実行したり、大きなデータを取り扱うことを可能にする EMS について解説しましょう(なお、ハードディスクについては、別途、4 章でくわしく解説します)。

3.1 ディスクの処理速度に関係する CONFIG.SYS ファイル

「CONFIG.SYS ファイル」(コンフィギュレーションファイル：MS-DOS システム構築ファイル)は、1章において、各種の日本語入力システム(FEP)を MS-DOS システムに組み込むために必要なファイルとして登場しています。そこでは、各種のデバイスドライバ(プログラム)を、システムに組み込むための指示について述べましたが、ここでもう一度、1.2 章(日本語入力システム ATOK の節)で使われている CONFIG.SYS ファイル(23 ページの図 1.12)をタイプアウトしてみましょう。

```
A>TYPE CONFIG.SYS⏎
FILES=10
BUFFERS=20
DEVICE=ATOK6A.SYS
DEVICE=ATOK6B.SYS

A>
```

1.2章で作成したATOKのCONFIG.SYSファイルをタイプアウトする

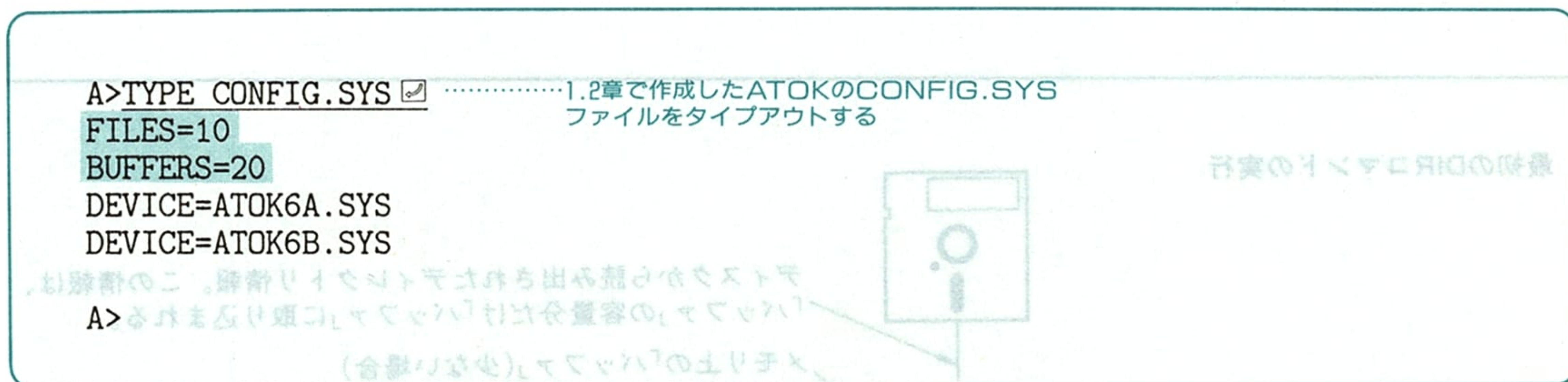



図 3.1　ATOK における CONFIG.SYS ファイルの一例

この CONFIG.SYS ファイルの内容の、次の部分に注目してください。

FILES＝10
BUFFERS＝20

この2行は1章では触れていませんが、MS-DOS のディスクの読み／書きに関するメモリ上の作業エリアの大きさを指定する行であり、「BUFFERS＝20」の行は、ディスクの読み／書きの処理速度に大きく関係します。本節では、CONFIG.SYS ファイルにおけるこの **BUFFERS 指定**について解説しましょう。

■ BUFFERS 指定によるディスクアクセスの高速化

たとえば、「BUFFERS＝20」とは、「バッファを 20 個分メモリ上に用意せよ」という指示を MS-DOS に与えるものです。この指示がない場合には、標準値として、MS-DOS バージョン 2.*x* の場合には「BUFFERS＝2」、MS-DOS バージョン 3.*x* の場合には、メインメモリの容量(SWITCH コマンドで設定した容量)によって次の値が設定されます。

384K バイト未満　……… 2
384K～511K バイト …… 5
512K～639K バイト…… 10
640K バイト以上 ……… 20

このバッファとは、ディスクアクセス(読み／書き)が行われる際のMS-DOS内部の作業のために、ディスクから読み出したディレクトリのデータなどを、メモリ上に一時的に保管しておくためのエリアのことをいいます。このバッファの数を多く取ると、一度のアクセスで、多くのデータを保管しておくことができるため、それ以降のディスクアクセスの回数を減らすことができます。これは〝ディスクバッファリング〟と呼ばれるMS-DOSの優れた機能の1つであり、ディスクの総合的な読み／書きの速度を大幅に改善することができます。

最初のDIRコマンドの実行
ディスクから読み出されたディレクトリ情報。この情報は、「バッファ」の容量分だけ「バッファ」に取り込まれる。
メモリ上の「バッファ」(少ない場合)
DIRコマンド実行
DIR
ディレクトリの表示
キー入力
コンピュータ本体

2度目以降のDIRコマンドの実行
「バッファ」が少ないため、前回ディスクから読み出されたディレクトリ情報の一部しか取り込まれていない。したがって再度ディスクがアクセスされる。
DIRコマンド実行
DIR
ディレクトリの表示
キー入力
再度ディスクから読み出されたものが表示される。

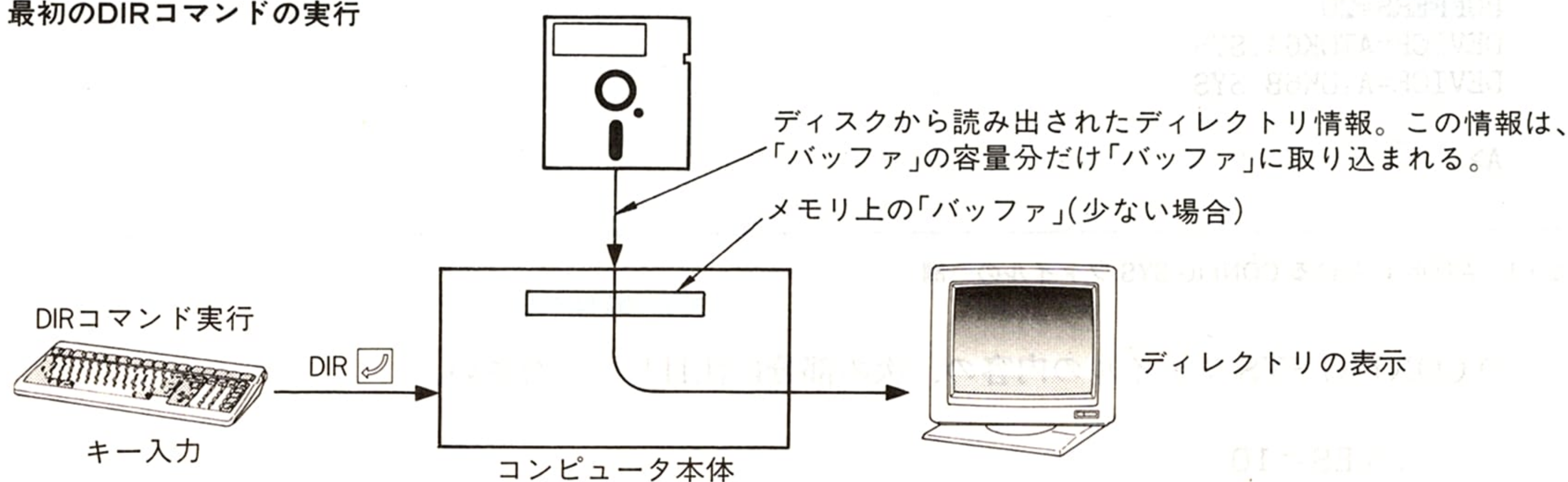
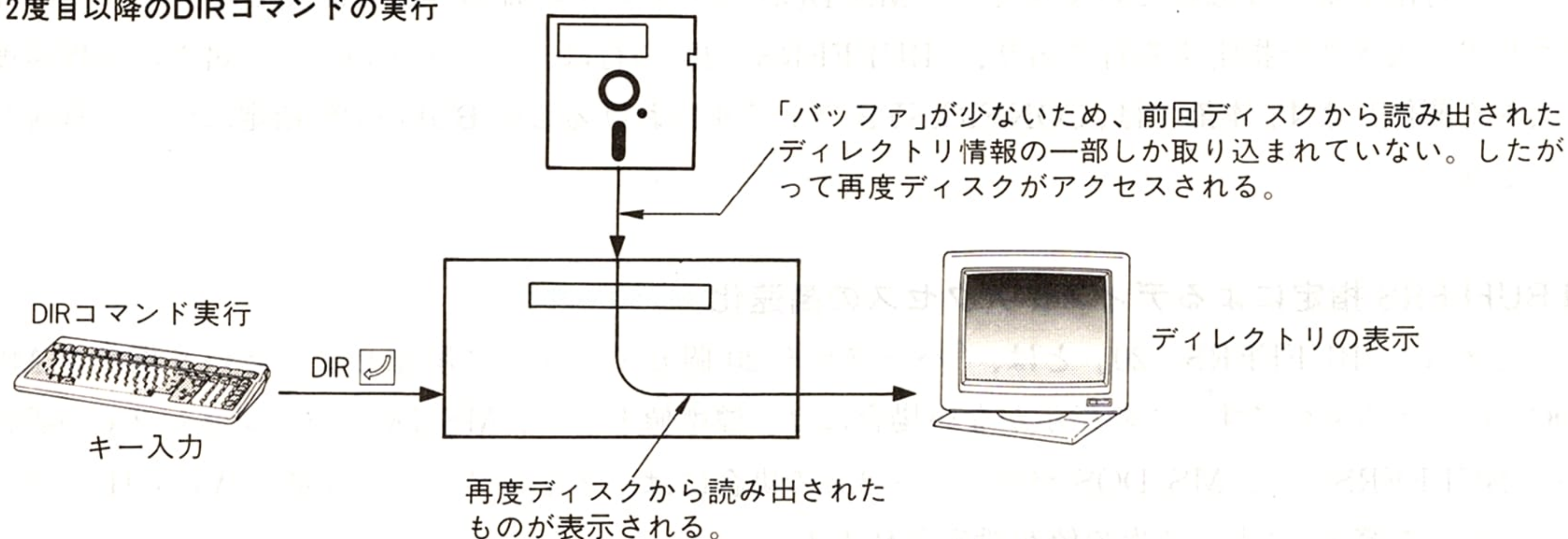
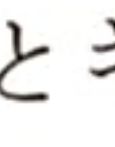
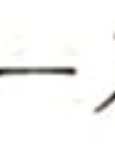
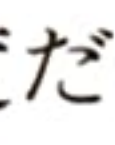



図3.2　DIR コマンドによるディスクバッファリング効果の確認(バッファ容量が小さい場合)

では実際に、このバッファの数を増やすことによる効果を、簡単な実験で確認してみましょう。まず、実験のために起動するMS-DOSのシステムディスク(ビジネスソフトのシステムディスクなどでもよい)のCONFIG.SYSファイル内の「BUFFERS=××」の数値を、「2」に変更してみてください。もし、このBUFFERSの行がない場合は、「BUFFERS=2」の1行を追加してください。また、CONFIG.SYSファイルそのものがない場合は、この1行のCONFIG.SYSファイルを作成してください。そして、「BUFFERS=2」のCONFIG.SYSファイルによって、MS-DOSを起動し直します。

さて、MS-DOSが起動したら、DIRコマンドを2〜3回実行してみてください。そして2回目以降も、「DIR ⏎」とキー入力するたびに、ディスクのアクセスが行われるかどうかを確認してください。アクセス時に点灯するディスクドライブの動作ランプや、ヘッドの動作音で確認することができるでしょう。「BUFFERS=2」の場合(MS-DOSバージョン2.*x*では、BUFFERS指定がない場合も)、「DIR ⏎」のキー入力のたびに、毎回ディスクがアクセスされるはずです。

これは、ディスクから読み出したディレクトリ(ディスク上のファイルの登録台帳)などのデータを、保管しておけるだけのディスクバッファの容量がないためであり、同じディスクであっても(ディスクの交換がされなくても)DIRコマンドが実行されるたびに、毎回同じように、それらのデータをディスクから読み出さなければならないのです。これではMS-DOS本来の、高速ディスクアクセスの機能は発揮できません。

ところが、このような状態のMS-DOSシステムディスクのCONFIG.SYSファイルのBUFFERS指定を、「BUFFERS=20」に変更してMS-DOSを再起動してみると、ディスクアクセスはどのように変化するでしょうか。それを実験してみましょう。ただし、旧型の5インチ2DDのフロッピーディスクドライブのなかには、残念ながらこのディスクバッファリングによる効果がないものがあります(PC-9800シリーズなど。後述)。

では、「BUFFERS=20」に変更したCONFIG.SYSファイルの内容を確認した後、リセットボタンを押して、MS-DOSを起動し直します。CONFIG.SYSファイルは、MS-DOSの起動時に自動的に読み出され、その内容で指示されたことが実行されますので、CONFIG.SYSファイルを変更した場合には、必ずMS-DOSを再起動する必要があります。

さっそくその効果を確かめてみましょう。以前と同様に、DIRコマンドを数回実行してみてください。最初の1度だけはディスクがアクセスされますが、その後の「DIR ⏎」のキー入力には、まったくディスクのアクセスなしにディレクトリが表示されるはずです。つまり、ディスクバッファの容量が十分大きいため、最初に読み出されたディレクトリのデータなどを全部そこに保管しておくことができ、したがって2度目以降は、ディスクをアクセスすることなく、ディスクバッファ上のデータを直接参照して、より速くDIRコマンドを実行できるわけです。

この効果は、ディスクに収容されているファイル数が多く、ディレクトリが深い(階層の層が多い)場合や、コマンドの多いバッチファイルを実行する場合などには、劇的な効果があります。

いったん読み出されたディレクトリのデータなどは、そのディスクが交換されるまで、「BUFFERS＝××」で確保されたメモリ上のエリアに存在しています。また、ファイルの新規作成や削除、更新などが行われた場合は、ディスク上のディレクトリが更新されるのと同時に、ディスクバッファ上のデータも同様に更新されます。しかし当然のことながら、別のディスクに交換されたときには、ディスクバッファ上のデータはただちに破棄され、そのディスクがアクセスされたときには、新しいディスクのデータに書き換えられます。言い換えれば、ディスクバッファリングの機能は、「ディスクが交換されるまで」(同じディスクが使われているうち)という大前提のもとに成り立つものです。もし途中でディスクが交換された場合は、そのことを確実に検知して、それまでのディスクバッファの内容をクリア(破棄)して、新しいディスクのデータに書き換えなければ、ディスク上のファイルや、処理データが致命的なダメージを受けることになります。

では、ディスクの交換を検知する機能を実験で確認してみましょう。方法は非常に簡単であり、さきほどから実験していたディスクドライブのドアを開け、ディスクをいったん取り出し(取り出さなくてもよいが)、それをそのままもとにもどしてドアを閉じ、再度DIRコマンドを実行してみてください。今までアクセスされなかったディスクが、今回はアクセスされたはずです。

これは、ディスクドライブのドアが開けられたことにより、ディスクの交換が行われたものとコンピュータが判断するためです。このようなことは、CTRL＋Cを入力した場合にも同じ状態が見られます。CTRL＋Cの入力により、MS-DOSシステムの各部が初期状態にリセットされ、同時にディスクバッファもクリアされるからです。

ただし、CONFIG.SYSファイルに登録するBUFFERSの数が、あまり多すぎると、かえって逆効果になりかねません。バッファ上のデータを処理する時間が無視できなくなるからです。フロッピーディスクでは、まあ20〜30程度、大容量のハードディスクでは、まあ50〜60程度でしょうか。指定できる数値は、2〜99の範囲です。

また、BUFFERSの指定により、ユーザープログラムが使用できるメモリエリアは、指定した数に応じたバッファのサイズだけ減少します。バッファ1個のサイズは、接続されているディスクのなかで、もっとも大きなセクタサイズを持つディスクの、約1セクタ分のメモリサイズであり、5インチや3.5インチ2HD、および8インチ2Dのフロッピーディスクであれば1040バイトです(1セクタは1024バイトですが)。またハードディスクの場合も、1セクタ1024バイトのものが多く、その場合、たとえば「BUFFERS＝20」を指定すると、約20Kバイトがディスクバッファのエリアとして確保され、その分ユーザーエリアが減少することになります。

ユーザーエリアの大きさが問題になるアプリケーションプログラムを実行する場合には注意が必要です。現在のアプリケーションプログラムのサイズや、それらが能率的に動作するために必要なメモリ容量を考えると、現在では、メインメモリはフル実装(640Kバイト)しておくことが常識です。

2度目以降のDIRコマンドの実行

メモリ上の「バッファ」を多く取ると、最初のディスクアクセスでディレクトリ情報がすべて「バッファ」に取り込まれるため、2度目以降はディレクトリに関するディスクからの読み出しを必要としない。つまりディスクアクセスは行われない

メモリ上の「バッファ」（多く取った場合）

DIR

キー入力

ディレクトリの表示

ディレクトリの全情報はメモリ上にあるので、ディスクをアクセスすることなく、メモリ内の処理により表示される。

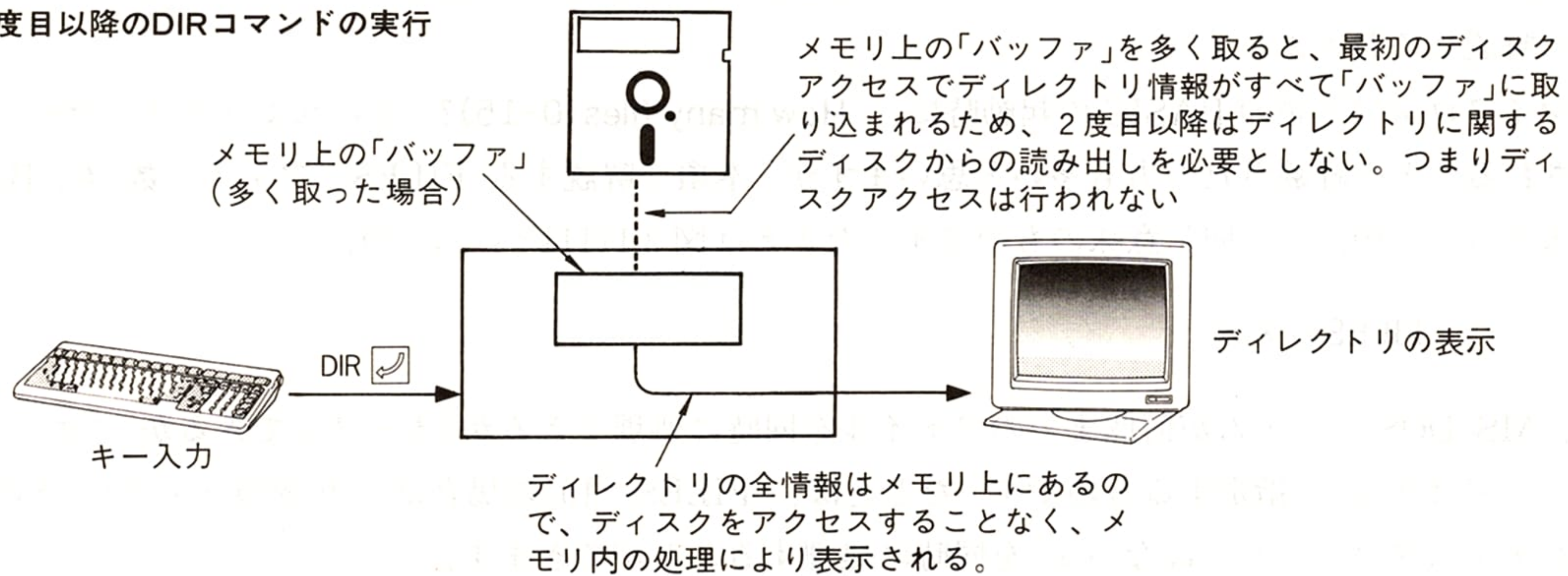



図 3.3 DIR コマンドによるディスクバッファリング効果の確認（バッファ容量が大きい場合）

さてここでは、DIR コマンドによる実験で、BUFFERS 指定によるディスクバッファリングの効果を確認しましたが、実際の作業においても、ディスクアクセスがともなうアプリケーションプログラムの実行では、大幅な処理速度の向上が見られます。ただしさきに述べたように、旧型の 5 インチ 2DD タイプのフロッピーディスクドライブのなかには、残念ながらこのディスクバッファによる効果がないものがあります。これは、それらのディスクドライブに、ディスクが交換されたかどうかをハードウェア的に検出する機能がない（あっても、それを使っていない）ことに起因しています。

普通のドライブは、閉っていたドアが開けられたことを検知して、ディスクの交換が行われたとみなしますが、その検知機能がないドライブでは、ディスクのデータを毎回実際に読み出してみないことには、交換されたかどうか判断ができません。そこで、ディスクの交換を検出できないドライブの場合は、ディスクの内容を比較してディスクの交換の有無を判断するなどのめんどうなことは行わず、ディスクをアクセスするたびに、毎回ディレクトリを読み出すことにして、ディスクバッファリングを行わないようにしているわけです。

このディスクバッファリングの機能は、MS-DOS の優れたファイル管理システムによるもので、MS-DOS はディスクアクセスが速いと言われている理由の 1 つです。いずれにしても、ディスクバッファリングの効果は、本項で行った DIR コマンドによる実験で、簡単に判定できますので試してみてください。このテストで効果がなかったディスクドライブやシステムは、残念ながらディスクバッファリングは働きません。この場合は、BUFFERS の数を多く指定しても意味はありません。ディスクバッファ用のメモリをむだに消費しないように、「BUFFERS=2」を指定しておいた方がよいでしょう（さきに述べたように、MS-DOS バージョン 3.*x* の場合は、メインメモリの容量に応じて、多くの BUFFERS が確保されるため）。

■ FILES 指定について

CONFIG.SYS ファイルに登録する **FILES 指定**は、処理速度には関係しませんが、これについても簡単に解説しておきましょう。

マイクロソフト系の BASIC の起動時に、「How many files (0-15)?」というようなメッセージが表示されることを経験された方も多いと思いますが、本項で解説する「FILES」(ファイル数)も、BASIC システムの「files」と同じ意味のものです。たとえば図 3.1(111 ページ)の、

FILES＝××

は、MS-DOS システムが何個までのファイルを同時に処理できるか(オープンできるか。「オープン」については後述)を指定するものです。たとえば、「FILES＝10」の場合は、10 個のファイル(そのすべてがディスクファイルではないが)を同時に処理することができます。

もし、CONFIG.SYS ファイルのなかで FILES 指定をしない場合は、標準値として、MS-DOS バージョン 2.*x* の場合は 5、バージョン 3.*x* の場合は 8 が設定されますが、多くのファイルを同時に処理するようなソフト(たとえば、データベースソフトや、ソフトウェアを開発する際のコンパイラやリンカなど)の実行には、この数が不足しないように注意が必要です。指定可能な数は、MS-DOS バージョン 3.*x* の場合、8～255 の範囲です。ただし、多くのファイルを同時に処理するような市販のアプリケーションプログラムには、あらかじめ「FILES」や「BUFFERS」が登録されている CONFIG.SYS ファイルが用意されているでしょう。

さて、MS-DOS などの OS(オペレーティングシステム)が、ディスクファイルをアクセス(読み／書き)する場合には、ディスク上のディレクトリ部(全ファイル登録台帳)に格納されている、目的のファイルの「ファイルコントロールブロック」(FCB)と呼ばれる「現住所登録簿」のデータを読み出し、メモリ上に用意しておかなければなりません。

このように、アクセスしようとするファイルの FCB のデータを、メモリ上に用意することを、「ファイルをオープンする」と言います。その逆に、オープン後、ファイルのいろいろな操作(変更、削除、新規作成など)によって変化したメモリ上の FCB の内容を、ディスクのディレクトリ部に書きもどすことを「ファイルをクローズする」と言います。ただし、この書きもどしのタイミングは、すべての処理の終了時だけでなく、処理の途中であっても、MS-DOS が自動的に行います。

つまり、さきほどの FILES 指定は、この「オープン」を、同時にいくつのファイルに対して行う必要があるか、その最大値を指示するものなのです。

ディスクをフォーマット処理する
FORMATプログラムを実行する場合
を想定している

全ファイル
登録台帳
各ファイルの内容収容庫
ディスクの内部
FORMAT
プログラム
の本体
ディスク側

ディスクのなかに
はフォーマットの
プログラムファイ
ル「FORMAT.EXE」
が存在している

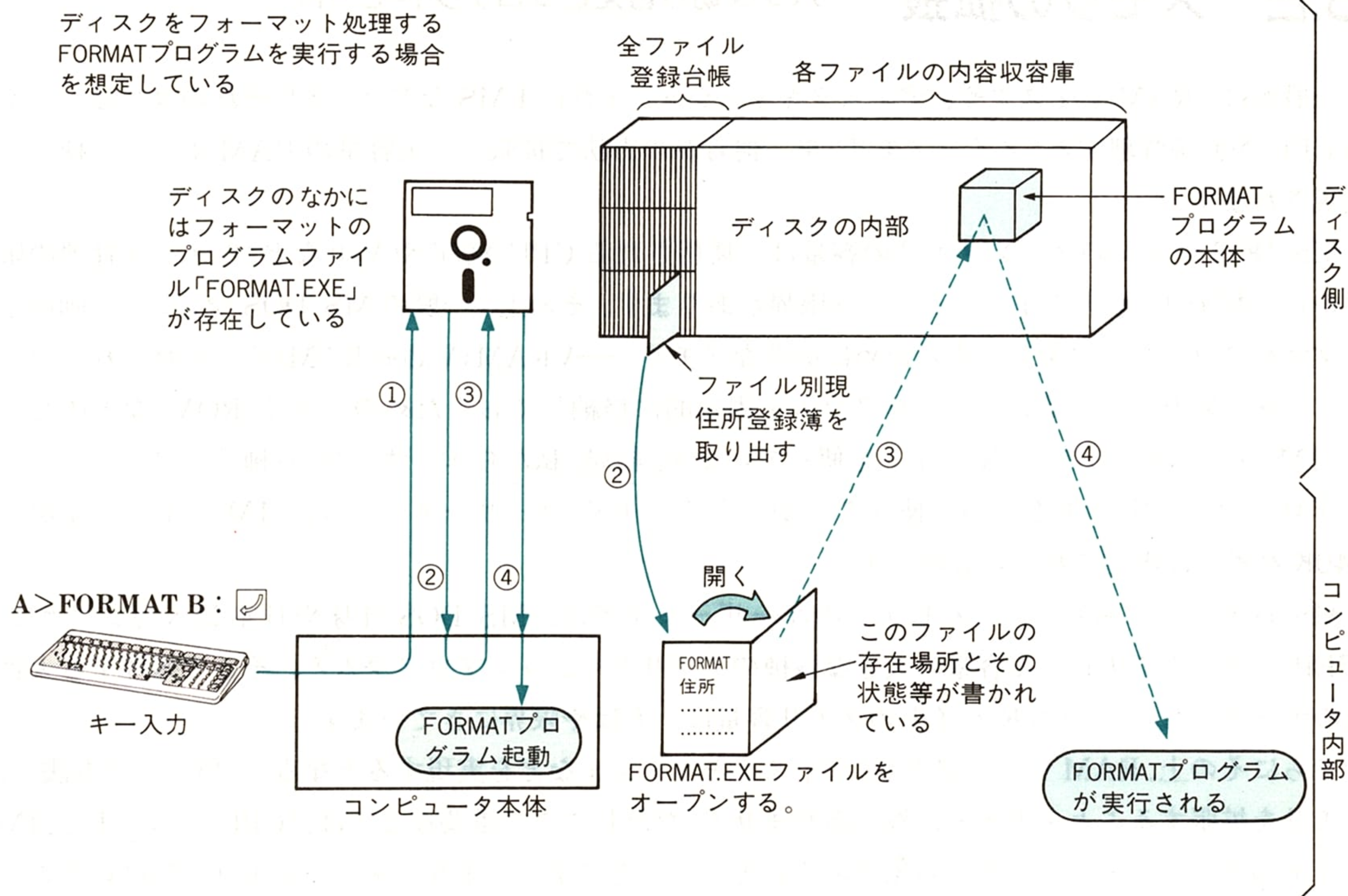
ファイル別現
住所登録簿を
取り出す

① ③ ② ③ ② ④ ④

A>FORMAT B：

キー入力
FORMATプロ
グラム起動
コンピュータ本体

開く
FORMAT
住所
………
………
このファイルの
存在場所とその
状態等が書かれ
ている
FORMAT.EXEファイルを
オープンする。

FORMATプログラム
が実行される
コンピュータ内部

① FORMATコマンドがキー入力されると、まずディスク上の「全ファイル登録台帳」(ディレクトリ部)を読みにいく。
② 目的のファイル「FORMAT.EXE」の「現住所登録簿」がメモリに取り込まれ、FORMATプログラムの存在場所とその状態が判明する。これを「FORMATプログラムのファイルがオープンされた」と言う。
③ FORMAT.EXEファイルの内容の所在が判明したので、その場所へFORMATプログラム自身を読みにいく。
④ FORMATプログラムが読み出され、メモリにロードされてFORMATプログラムが実行される。

☆ ファイルの「クローズ」はこれの逆の動作で、ファイルの内容の変更等のために書き直された「現住所登録簿」を、ディスク上にもどすことを言う。

図 3.4　ファイルのオープンとクローズ

3.2 メモリの拡張 ―バンク切り替えとプロテクトモード―

一般的に、RAM ディスクや、ディスクキャッシュ、それに EMS などは、それぞれのコンピュータの CPU が直接管理する「メインメモリ」を、何らかの方法で拡張した大容量の RAM ボードを使って実現されます。

MS-DOS マシンのメインメモリの容量は、使用される CPU(8086 や V30 など)のメモリ管理の限界から、本質的に最大 1M バイトという限界があります。その上、一般の MS-DOS マシンは、画面上に文字やグラフィックを表示するために必要なメモリ――VRAM(Video RAM)や、それぞれのコンピュータに固有のシステム的なプログラムを固定的に格納しておくためのメモリ(ROM)などのために、1M バイトのうちの 384K バイトを使っています。結局、私たちユーザーが、各種のアプリケーションプログラムを実行するために使うことができるメモリ(ユーザーエリア)は、1M バイトのなかの **640K バイト**に限定されているのです。

MS-DOS マシンのメインメモリ上のユーザーエリアは、MS-DOS 自身や日本語入力システム(FEP)、その上で実行する容量の大きな各種のアプリケーションプログラムと、そこで処理する大容量のデータのために、640K バイトのメモリ容量は、もはや限界にきています。

さらにその上、RAM ディスクとか、ディスクキャッシュなどを実現するとなると、何らかの方法で、メモリを拡張することを考える以外にありません。ただしここで重要なことは、『CPU はあくまで、1M バイトのメインメモリが処理の対象である』ということです。つまり、メインメモリの外にいくらメモリを継ぎ足しても、そこのデータを CPU が直接処理するわけにはいきません。そこで**バンク切り替え方式によるメモリの拡張**とか、**プロテクトモードによるメモリの増設**などが考え出されました。

■ バンク切り替え方式によるメモリの拡張

まず、MS-DOS マシンにおける、一般的なバンク切り替え方式の概念を**図 3.5** で示しましょう。

この図のように、ユーザーが使用できる 640K バイトのメインメモリは、128K バイトの 5 つのブロックから構成されています。その最後(メモリの番地が高い方)の 128K バイトのブロックが、バンク切り替え可能なメモリブロックであり、このブロックの背後に、同じく 1 枚が 128K バイトのメモリのブロックがズラッと並べられています。これが**バンクメモリ**です。

CPU が処理できるのは、ある瞬間ある瞬間を見ると、あくまでこれらのバンクメモリのなかのいずれか 1 枚であり、その瞬間に選択されている 1 枚のメモリが、連続した 640K バイトのメモリの、最後の 128K バイトとして機能します。実行中のプログラムの必要に応じて、バンクメモリのなかのどの 1 枚を「生かす」か、それを瞬間瞬間に切り替える方式が**バンク切り替え**と呼ばれるものです。バンクメモリのなかの、選択されたもの以外は、いわば「死んで」いるメモリであり、CPU から見ると、それらは存在していないわけです。

CPUに8086やV30を使ったマシンの場合

0
640K
1M
メインメモリ最大1Mバイト
一般的なMS-DOSマシンのユーザーエリア
128Kバイト
グラフィック用VRAM
テキスト用VRAM
システムROM
などに、マシンが使用しているエリア
ある時点を見ると、これらのバンクメモリのなかの1枚(128Kバイト)がメインメモリとして動作状態にある。
バンクメモリ
増設した2MバイトのRAM
メインメモリのこの128Kバイト分が、増設されたバンクメモリと切り替えられる(バンク切り替え)。したがって、この部分の128Kバイトのメインメモリは、バンク切り替え可能なメモリボードでなければならない。

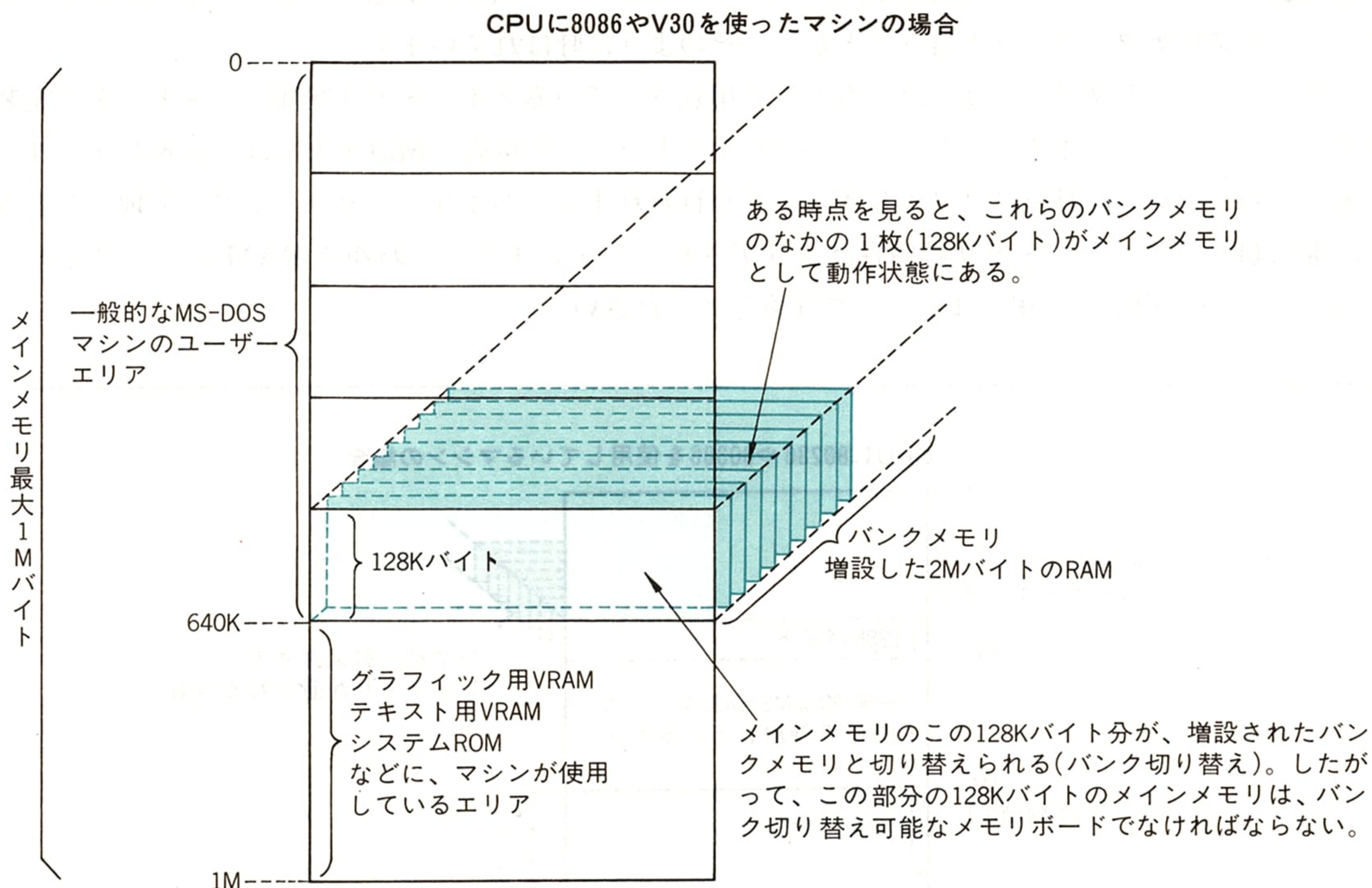



図3.5　バンク切り替え方式の概念

本章3.6で解説するEMSも、考え方はこのバンク切り替え方式によるメモリと同じであり、ただバンクメモリを置く番地や、バンクの構造などが異なるだけです。

■ プロテクトモードによるメモリの増設

CPUに8086やV30を使ったMS-DOSマシンでは、前項のバンク切り替え方式に頼る以外に、メモリを拡張するよい手段はありませんが、80286や80386というCPUを使ったMS-DOSマシンでは、「プロテクトモードによるメモリの増設」という方法があります。

80286というCPUは、16ビットCPUの8086と、32ビットCPUの80386の中間に位置するもので、16ビットCPUでありながら、メインメモリだけは、仮想メモリ方式により16Mバイトを管理する機能があります(80386は仮想メモリにより4Gバイトまで管理可能)。つまり、8086やV30では最大1Mバイトであったメインメモリを、それ以上の番地にまで広げることができるわけです。

この80286や80386のCPUの特徴を活かして、MS-DOSのメインメモリを増設する手段が**プロテクト増設メモリ**です。80286や80386CPUにおいて、1Mバイト以上の番地のメインメモリを使うためのモードを**プロテクトモード**と言うことから、そのように呼ばれています。

この「プロテクト増設メモリ」は、もともと用意されているメインメモリ空間に、メモリを「実装する」という形になります。ただし、このプロテクトモードの場合、増設メモリは「正規のメモリ」であるため、マシン起動時にはメモリチェックが行われます。つまり、リセットボタンを押したときや、電源ON時に、そのメモリの内容はクリアされてしまいます。いわゆるRAMディスクなどの「ウォームブート機能」は働きませんので注意してください。

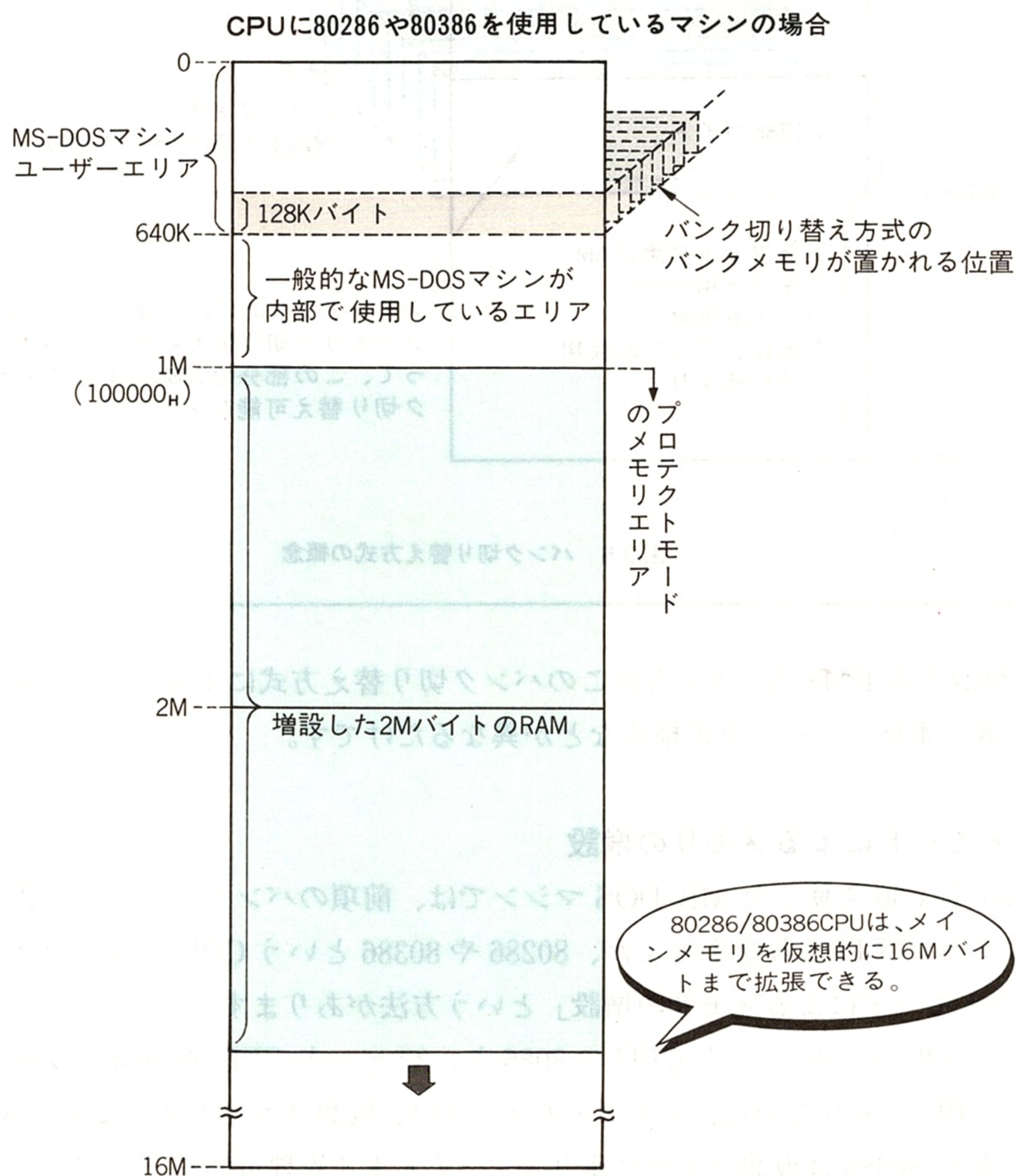



図3.6　プロテクトモード増設メモリの概念

さて、現在の 16 ビット MS-DOS マシンでは、実行するアプリケーションプログラムが発展するにともない、そのプログラムの容量はますます大きく、データを処理するための作業エリアもどんどん拡大する一方です。そこで後述の **EMS** というメモリ拡張方式が提唱されました。それは MS-DOS マシンには有効な方法であり、640K バイトの限界を一挙に 8M バイトとか 16M バイトとかに広げる効果はあるものの、やはり本格的な対応には、「正規のメインメモリ」を拡大できる 32 ビットマシンとそれに対応した OS に期待しなければならないでしょう。

バンク切り替え方式は、バンクを切り替えるための処理や、バンクメモリ上のデータを管理するための処理が負担になります。また、プロテクトモード増設メモリの場合においても、リアルモード(通常のモード)から増設メモリを特別な方法でアクセスしたり、そのメモリ上のデータを 640K 内のメインメモリ上に転送するための処理が大きな負担になります。結局いずれの場合も、通常のメインメモリ上での処理に比べ、処理スピードの低下となって表れます。

3.3　多目的 RAM ボードと統合型ドライバソフト

RAM ディスクやディスクキャッシュ、それに EMS などを実現するには、増設 RAM ボードと、その RAM ボード上でそれぞれの機能を実現させるためのソフトウェアが必要です。最近の RAM ボードは、メモリ用 LSI の高集積化と価格の低下にともなって大容量化が進み、1 ボードで 1～4M バイト程度のものは常識となり、さらに大容量のものまで使われています。

さて、1 枚の RAM ボードが数 M バイトの容量を持つようになると、1 つの RAM ボードを単独の機能としてではなく、内部的に分割して、**RAM ディスク**や**ディスクキャッシュ**、それに **EMS** など、いろいろな用途に同時に使うことができるようになりました。最近の各メーカーの RAM ボードは、たいていそのような、同時に複数の機能を実現できるようになっています。

また、このような RAM ボードをコントロールするソフトウェアも、同時に複数の機能を実現したり、それらのさまざまな使い方に対応するために、各種の機能を統合した形態のものが多くなってきました。

■ RAM ディスク／ディスクキャッシュ／EMS、統合型ソフト「MELWARE（メルウェア）」

RAM ディスクやディスクキャッシュ、それに EMS を実現するための RAM ボードや、そのソフトウェアの一例として、メルコ社の RAM ボード「HCE-2000」と、それに付属のソフトウェア **MELWARE** を取り上げます(ここではその PC-9800 シリーズ対応のものを使用する)。ただしこの MELWARE は、最近の一般の RAM ボードのほとんどに対応するソフトウェアですので、RAM ボードはとくに HCE-2000 に限定する必要はありません。

最近の多くのRAMボードがそうであるように、「HCE-2000」(容量2Mバイト)も、CPUに8086やV30を使った機種では、バンク切り替え方式によるバンクメモリとして動作し、CPUに80286や80386を使った機種では、バンク切り替え方式、またはメモリアドレス100000H以上(1Mバイト以上)のプロテクトモード増設メモリとして動作します(このことは前節で解説した)。いずれの場合も、メインメモリは640Kバイトのフル実装が前提です。

このRAMボードは、付属の統合型ソフト「MELWARE」によってコントロールされ、バンクメモリ上、あるいはプロテクトモードの増設RAM上に、RAMディスクや、後述のディスクキャッシュ、それにEMSによる拡張メモリを実現します。

ではまず、MELWAREのオリジナルディスクに含まれている各種のファイルを、DIRコマンドで示します。

```
A>DIR

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

MELCO232 LOD     16384  88-12-29   11:48
MELCO233 LOD       471  88-12-22   20:32
MELEMM   SYS     16416  89-02-21   11:43
MELCACHE SYS      9253  89-01-18   15:02
MELDISK  SYS      5888  89-01-05   22:00
FPLD     SYS      3820  89-02-17   19:29

AUTOEXEC BAT        60  88-01-31    0:40
MELTEMP  BAT      1739  89-02-21   18:44
MELCO    BAT       196  89-01-05   11:05
MANUAL   DOC     38236  89-01-26   18:22
89022100             2  89-02-21   11:54
        33 個のファイルがあります.
    972800 バイトが使用可能です.

A>
```

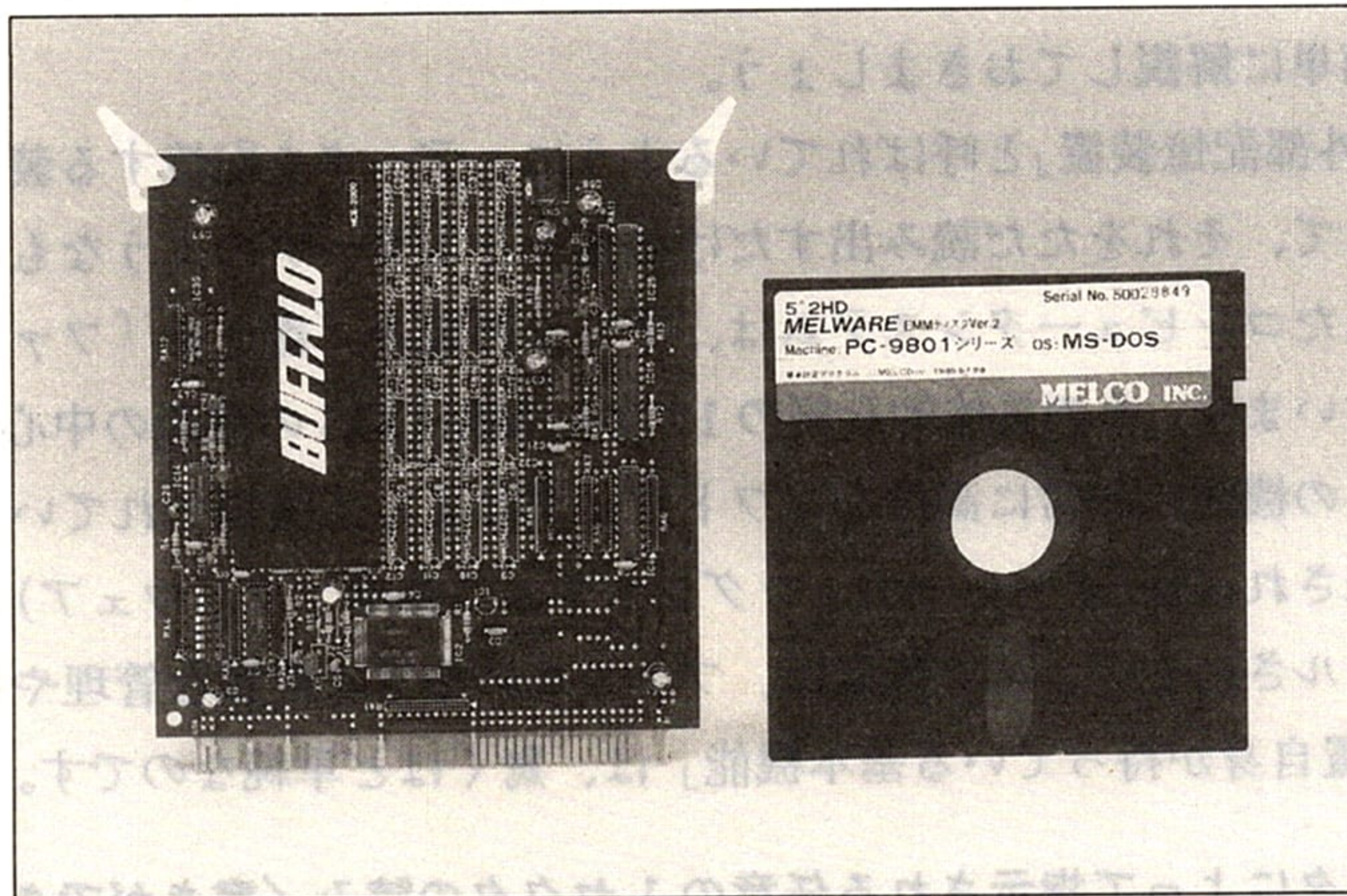
EMMドライバ(EMSドライバ+バンクメモリ管理マネージャ)。すべての場合に必須
ディスクキャッシュドライバ
RAMディスクドライバ
簡単設定プログラムを使って、初期設定を行うためには、このなかのいくつかのファイルが必要である
メニュー方式による簡単設定用バッチファイル

図3.7　MELWAREのオリジナルディスクに含まれる各種のファイル

これらのファイルのなかで、RAMディスクを実現するためのプログラム(デバイスドライバ形式)は、MELDISK.SYSですが、これが動作するには、MELEMM.SYSも必要です。「MELEMM.SYS」は、EMSを実現するためのデバイスドライバですが、RAMディスクやディスクキャッシュのデバイスドライバが動作するためには必須のプログラムです(そのなかの一部――バンクメモリのバンクを管理する機能の部分が必要)。また、ディスクキャッシュを実現するためのデバイスドライバはMELCACHE.SYSです。

結局、RAM ディスクやディスクキャッシュ、EMS を実現するために、最低限必要なプログラムやファイルは、次のようになります。

● RAM ディスク
「MELEMM.SYS」+「MELDISK.SYS」+「CONFIG.SYS」
●ディスクキャッシュ
「MELEMM.SYS」+「MELCACHE.SYS」+「CONFIG.SYS」
● EMS
「MELEMM.SYS」+「CONFIG.SYS」
● EMS+RAM ディスク+ディスクキャッシュ
「MELEMM.SYS」+「MELDISK.SYS」+「MELCACHE.SYS」+「CONFIG.SYS」

つまり、EMS のデバイスドライバだけは、いずれの場合にも必要であり、あとは実現するそれぞれのデバイスドライバが存在していればよいわけです。なお実現する機能は、CONFIG.SYS ファイルに「DEVICE=xxxx」のように、それぞれのデバイスドライバの名前を登録しておかなければなりません。実際の登録内容は、以降の節で示しますが、その CONFIG.SYS ファイルの指示により、それぞれの機能が、MS-DOS の起動時に MS-DOS システムに組み込まれます。なおデバイスドライバについては、1.1 章や本章の 3.1 を参照してください。

2M バイトの RAM ボード（HCE-2000）と統合型ドライバソフト（MELWARE）
ーメルコ社ー

3.4 RAMディスク

ハードディスクは、フロッピーディスクと比較して、ディスクアクセスが格段に速く、これを利用することにより、全体の処理速度を大幅に改善することができます。しかし、さらに高速の「ディスク装置」としては、純電子的に構成された**RAMディスク**があります。RAMディスクは、RAMボードとRAMディスクソフトにより簡単に実現することができ、操作も通常のディスクと同じであるため、スピードを要求するいろいろな処理に広く使われています。

本節では、RAMディスクと呼ばれている「ディスク装置」について解説し、MELWAREを使って、実際にRAMディスクを実現してみましょう。

■ RAMディスクとは

RAMディスクとは、通常のディスク装置の記録媒体である磁気円盤を、純電子的なLSI(高密度集積回路)で構成されたRAM(コンピュータのメモリなどに使われているランダムアクセスメモリ)に置き換えたもので、コンピュータ側から見ると、通常のディスク装置とまったく同じ扱いで、同じ動作をするようにつくられた超高速ディスク装置です。つまり、ディスクとはいっても物理的な円盤が存在しているわけではなく、通常のディスク装置と同じ動きをするように、RAMを使って実現した**仮想ディスク装置**であるわけです。

RAMディスクの話を進める前に、まず、フロッピーディスクやハードディスクなどのディスク装置に関する基礎知識を簡単に解説しておきましょう。

ディスク装置は、「外部記憶装置」と呼ばれているように、データを記憶する装置ですが、単にデータをズラズラと記憶して、それをただ読み出すだけのカセットテープのようなものではありません。ディスク装置を接続したコンピュータシステムは、データの読み／書きを「ファイル」という形式で管理する機能をもっています。その具体的な例の1つが、MS-DOSの機能の中心である、ファイル管理システムですが、この機能は非常に高度なソフトウェアによって実現されています。

コンピュータに接続され、OS(オペレーティングシステム：基本ソフトウェア)の高度なソフトウェアによってコントロールされるディスク装置は、ファイルの非常に複雑な管理や操作を行います。ところが、「ディスク装置自身が持っている基本機能」は、驚くほど単純なのです。その基本機能とは、

コンピュータによって指示される任意の1セクタの読み／書きができる

ということだけなのです。ディスク装置自身には、ただこれだけの機能しか備わっていません。ディスク装置のこの機能だけを利用して、あとはすべてソフトウェアによって高度なファイル管理システムが実現されているわけです。

つまり、ディスク装置自身にとって、「ファイル」とか「ファイル管理」などの概念はまるでなく、

ディスク上に記録されるディレクトリ部も、階層的ディレクトリも、ファイルも知ったことではありません。ただ指示された「あるトラックのあるセクタを機械的に読み／書きする」だけなのです。これらのことを次の図で示しましょう。

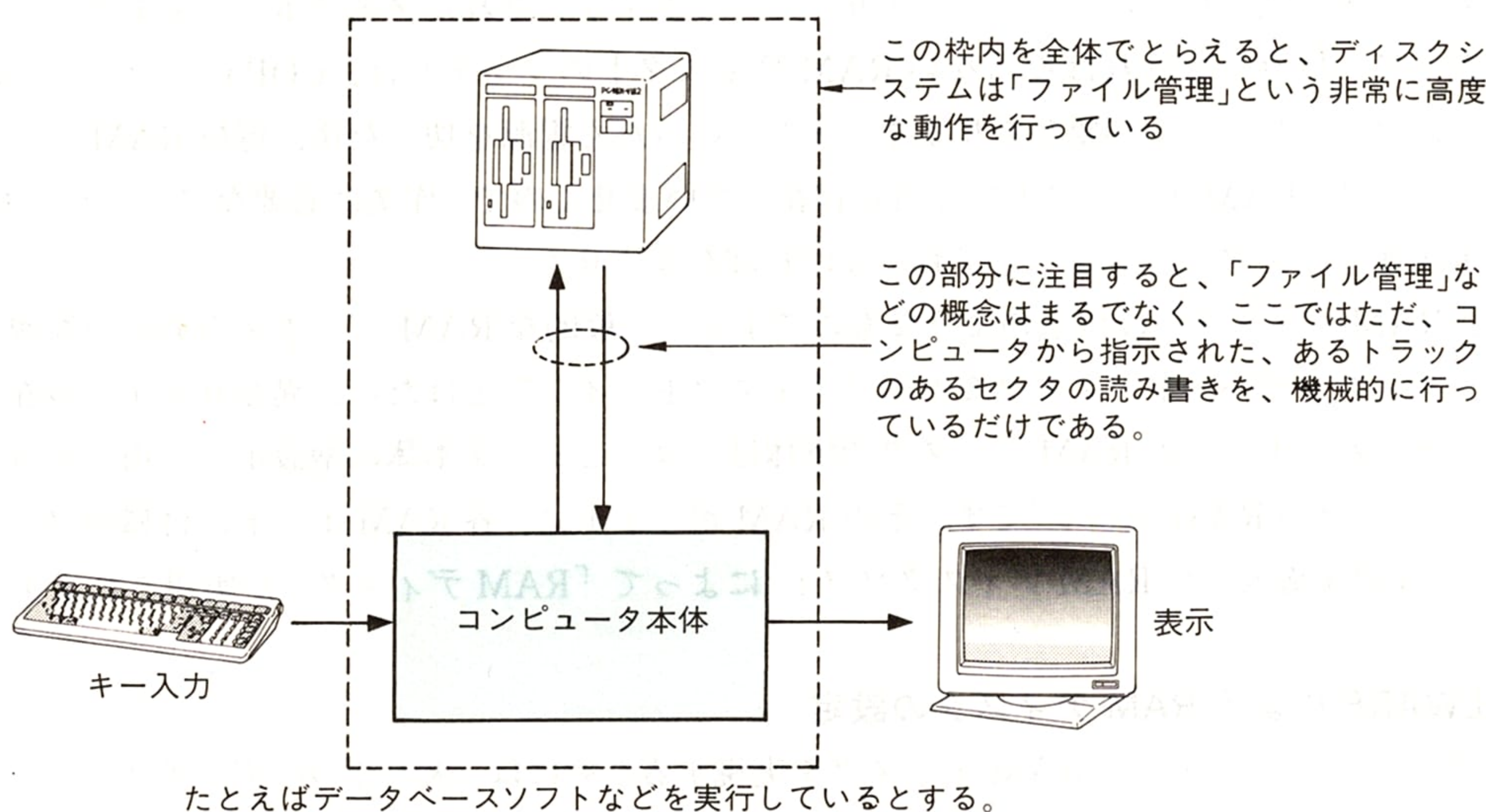



図 3.8 ディスク装置の基本機能と、ファイル管理機能

ディスク装置のあらましはこのようなものですが、このディスク装置と等価の機能を RAM で実現したものが「RAM ディスク」です。では、RAM ディスクに話を移しましょう。

RAM ディスクと同じ概念の装置が、パーソナルコンピュータ上で実現されたのは意外と古く、1981年頃、8 ビットのパーソナルコンピュータの初期に、FM-8 のバブルメモリ*ですでに実用されています。

しかしバブルメモリは、読み／書きの速度が遅く、また、メモリ容量 対 容積比も悪いため、不揮発性メモリである利点にもかかわらず、広く普及するには至りませんでした。ところが最近では、メモリ素子(RAM チップ)の高密度化、大容量化、低価格化が進み、RAM を使った大容量の仮想ディスクが容易に実現できるようになり、いわゆる「RAM ディスク」が広く使われるようになりました。

* バブルメモリ：通常の RAM が、データを「電荷」によって記憶し揮発性であるのに対し、このメモリは、「磁気の泡」によって記憶し不揮発性である。

RAMディスクは磁気円盤のような可動部分がなく、純電子的に構成され、読み／書きの速度はRAMの読み／書きの速度で行われますので超高速です。このRAMディスクを使うための各種のコマンドのキー操作は、とくにRAMディスクであることを意識する必要はなく、通常のディスク装置と同じです。ただし、記録媒体が揮発性のRAMであるため、電源を切ると、記録されているデータがすべて失われてしまう点が通常のディスク装置と決定的に異なりますので注意が必要です(電源を切ってもデータが失われないようにバッテリーでバックアップされたメモリボードもある)。

したがって、保存しなければならないRAMディスク上のファイルは、COPYコマンドで通常のディスクにコピーしておく必要があります。また、いったん電源を切った後、再びRAMディスクを使用する場合は、RAMディスク上には何も存在していませんので、作業に必要なファイルがあれば、事前にRAMディスク上にコピーしておかなければなりません。

さて、RAMディスクとは以上のようなものですが、一般的なRAMディスクの実際の外観は、フロッピーディスクやハードディスクなどの〝ディスクドライブ〟とはだいぶ異なります。現在使われている、ほとんどすべてのRAMディスクの実体は、コンピュータ本体の増設ボード用のスロットに挿入された〝ただのRAMボード〟です。そのRAMボード上に、各RAMボードに付属のソフトウェア(あるいは別途購入したRAMディスクソフト)によって「RAMディスク」が実現されます。

■ MELWAREによるRAMディスクの設定

本章の3.3で述べたように、RAMディスクを実現するためには、次の2つのデバイスドライバのプログラムファイルが必要です。

- MELEMM.SYS　………　EMMドライバ(本章3.6で解説)
- MELDISK.SYS　………　RAMディスクドライバ

これらのデバイスドライバを動作させるためには、CONFIG.SYSファイルへ次の2行を登録しておく必要があります。

```
例：　DEVICE = MELEMM.SYS  /U0
      DEVICE = MELDISK.SYS  B  1920  /S
```

これらのデバイスドライバは、任意のディレクトリ上に置くことが可能ですが、その場合は、それぞれの「.SYS」ファイルにパス名を付ける必要があります(たとえば「DEVICE=¥RAM¥MELEMM.SYS /U0」というように)。

「DEVICE=MELEMM.SYS /U0」の行は、EMSを設定する行ですが、RAMディスクやディスクキャッシュを設定する前提として、EMSを使わない場合にも必要です。この「/U0」の部分はオプションパラメータであり、EMSを実現しない指定です。つまり、MELEMM.SYS内の、バンク管理の機能の部分のみを使用する場合に指定します。ここではEMSを使いませんので、「/U0」を指定し

ておきます。なお EMS の設定については、本章の 3.6 でくわしく解説します。

次の行、「DEVICE＝MELDISK.SYS B 1920 /S」の行は、RAM ディスクを設定する行です。「B 1920 /S」の部分は、オプションパラメータの一例ですが、その一般的な書式を次に示します。

DEVICE ＝ MELDISK.SYS (B)(メモリ容量)／(ルートファイル数)(S)

- ●バンクメモリのみを使う場合は「B」(プロテクトモードの場合は「B」を付けない)
- ●「メモリ容量」は確保する RAM ディスクの容量(K バイト単位で指定。省略した場合は増設ボードの全メモリ)
- ●「ルートファイル数」はルートディレクトリに収容可能な最大ファイル数(32 の倍数で指定。最大は 512。省略した場合は 128)
- ●「S」を付けると、RAM ディスクのメモリのサムチェック(読み出し時のデータチェック)を行う(省略した場合はサムチェックなし)

たとえば、「DEVICE＝MELDISK.SYS B 1920 /512 S」は、バンクメモリの 1920K バイトを RAM ディスク領域として確保し、ルートディレクトリの最大ファイル数が 512 で、サムチェックを行う RAM ディスクを実現する指示となります(1920 という値は、2M バイトの RAM ボードを〝バンク切り替え方式〟で使用する場合のもので、2M、つまり 2048K バイトのなかの 128K バイトは、メインメモリとして使われるため、RAM ディスクメモリはその 128K バイト分減少します)。

また、メニュー方式により、RAM ディスク／ディスクキャッシュ／EMS の各設定を簡単に行うことができる「簡単設定プログラム」が用意されていますが、これについては本章の 3.7 にまとめて示します。

では、1.4 章で使った日本語入力システムの VJE を組み込んだシステムディスク上に、RAM ディスクを設定する一連の実行例を示しましょう。ここではバンク切り替えの RAM ボード上に(つまりプロテクト増設メモリ上ではなく)RAM ディスクを実現します。

MELWARE のオリジナルディスク(またはそのバックアップコピー)をドライブ A：に、VJE を組み込んだシステムディスクをドライブ B：にセットして作業を始めます。

```
A>COPY MELEMM.SYS B:⏎
        1 個のファイルをコピーしました.

A>COPY MELDISK.SYS B:⏎
        1 個のファイルをコピーしました.

A>DIR B:⏎

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

COMMAND  COM     24931  88-10-16   16:10
VJEB     SYS     76913  89-01-31    2:00
VJEB     DRV      2763  88-10-10    2:00
VJEB     DIC    488448  89-01-08    2:00
SETVJE   EXE     15394  88-11-01    2:00
CONFIG   SYS        84  88-09-19    2:00
MELEMM   SYS     16416  89-02-21   11:43
MELDISK  SYS      5888  89-01-05   22:00
        8 個のファイルがあります.
   519168 バイトが使用可能です.

A>

B>TYPE CONFIG.SYS⏎
FILES=10
BUFFERS=20
DEVICE = MELEMM.SYS /U0
DEVICE = MELDISK.SYS B 1920 /S
DEVICE = VJEB.DRV /HB /M1 /T0 /L /J /K- /K, /K. /K[ /K/ /D4

B>
```


図 3.9　VJE を組み込んだ MS-DOS のシステムディスク上に RAM ディスクを設定する

なお、CONFIG.SYS ファイルの書き換え(2 行追加)は、エディタやワープロを使って行いますが、COPY コマンドを使って、全体を作り直してもかまいません。

以上のような設定をした後、このシステムディスクで MS-DOS を再起動すれば、RAM ディスクが使用可能となります。では、RAM ディスクが設定されたシステムディスクをドライブ A：にセットして、リセットボタンを押してみましょう。

■ RAM ディスク上でのユーザープログラムの実行

さて、RAM ディスクが設定されたシステムディスクを起動すると、MS-DOS のオープニングメッセージとともに、RAM ディスクなどのデバイスドライバがシステムに組み込まれ、その際の各種のメッセージが表示されます。

```
(リセットボタンを押す)
NEC PC-9800 Series Personal Computer
ﾏｲｸﾛｿﾌﾄ MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30
Copyright 1981,88 Microsoft Corp. / NEC Corporation
                                        } MS-DOSのオープニングメッセージ

   》バンクメモリーマネジャー《      [Ver2.21] Jan.1989 MELCO Inc. Japan

ＲＡＭディスクドライバー 《ＭＥＬ_ＤＩＳＫ》を組み込みました。
[Ver 2.10] Copyright 1989  MELCO.Inc Japan
データチェックにサムチェックを行います。
ドライブは[ D: ]で[ １９２０ＫＢ]を割当ました。
        } EMMドライバ／RAMディスクドライバの組み込みメッセージ。1920KバイトのRAMディスクがドライブD：に設定されている

日本語入力フロント・プロセッサ ＶＪＥ－β 第２.００版
Copyright (C) 1986-88 VACS Corp. / ASCII Corp.
        } VJE-βのFEPドライバの組み込みメッセージ

Command ﾊﾞｰｼﾞｮﾝ 3.30
現在の日付は 1989-09-06（水）です.
日付を入力してください:⏎
現在の時刻は 21:06:10.00 です.
時刻を入力してください:⏎

A> ……………システムが完全に立ち上がった。ドライブD：において、1920Kバイトの容量のRAMディスクが使用できる
```

図 3.10 RAM ディスクが組み込まれたことを示す MS-DOS 起動時のメッセージ

このメッセージには、RAM ディスクに割り当てられたドライブ名が「D：」であり、そのメモリ容量が 1920K バイトであることなどが表示されています。つまり、ドライブ D：として、容量 1920K バイトの RAM ディスクが実現されたわけです。RAM ディスクが割り当てられるドライブ名は、通常のディスクドライブが割り当てられているドライブ名の次のドライブに、自動的に割り当てられます。

RAMディスクのドライブ名は、実際に存在するドライブのつぎに割り当てられる

(　)は、ハードディスクから起動した場合のドライブ名

ハードディスク

A：(B：)

C：(A：)

B：(C：)

RAM ディスク

D：

この構成の場合は、実際に存在するディスクドライブ「C：」のつぎのディスクドライブ「D：」に自動的に割り当てられる。

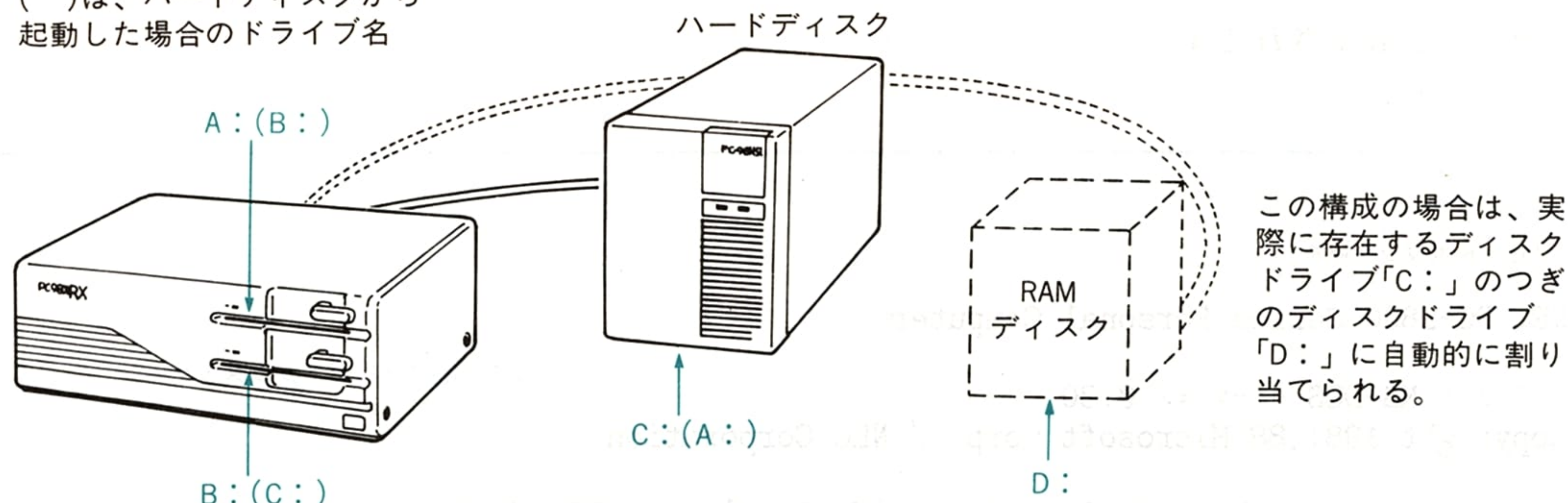
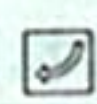
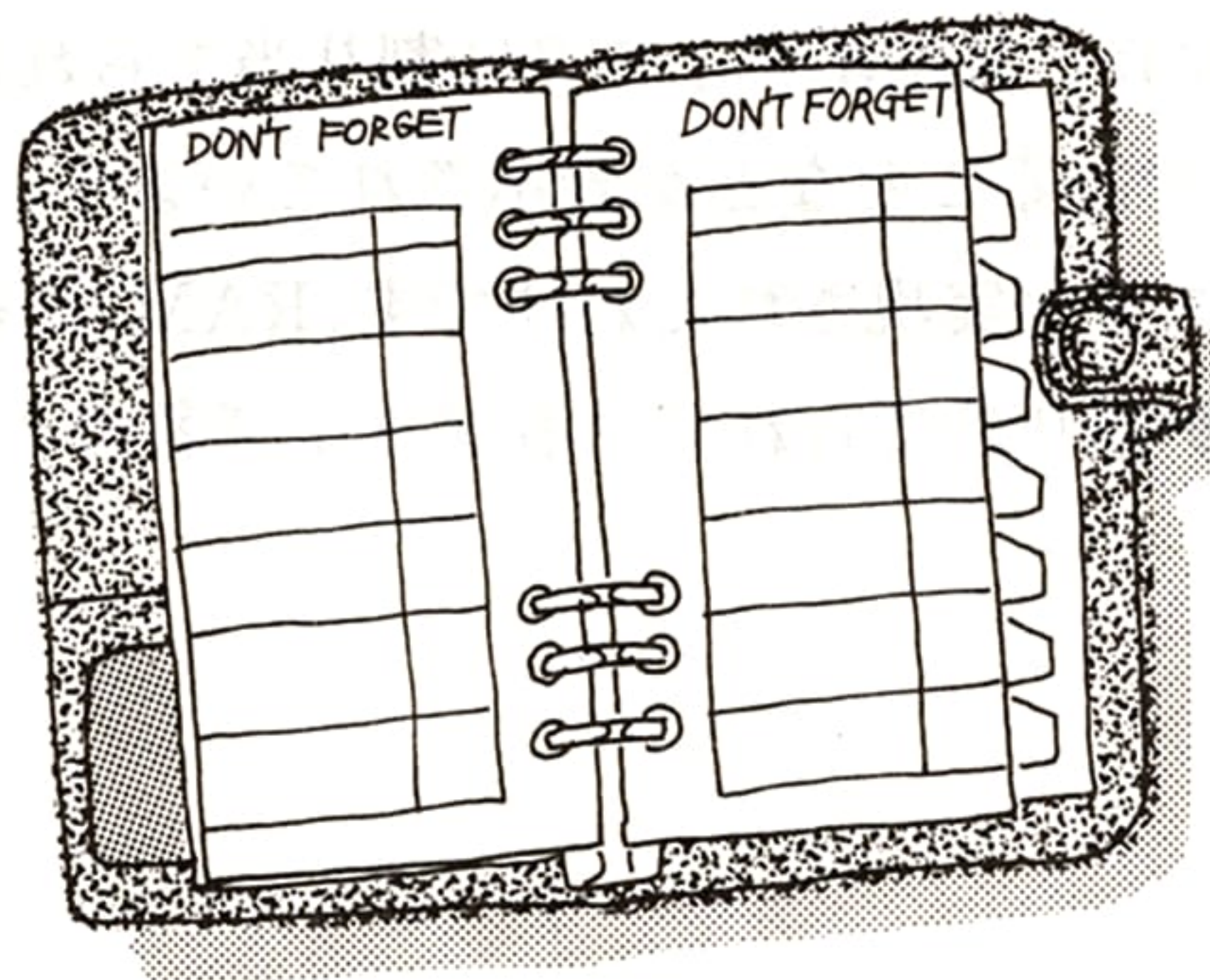



図 3.11　RAM ディスクのドライブ名の割り当て

RAM ディスクの威力は、ワープロや、データベースソフト、それにアセンブラやコンパイラなどの、ディスクアクセスが頻繁に行われるソフトウェアの実行において劇的に発揮されます。ここではそのことを、簡単な実験で確認してみましょう。

最初の実験は、現在起動しているシステムディスクの FEP である VJE を使って、文節変換などを RAM ディスク上で行ってみます。そのためには、まず VJE の辞書ファイル「VJEB.DIC」を、RAM ディスクであるドライブ D：上(接続されているディスクドライブの構成によって異なる)にコピーしておく必要があります。RAM ディスクも、取り扱いは通常のディスクとまったく同じですので、いつもと同様に COPY コマンドを実行します。そのあと、VJE 内の、辞書ドライブの変更機能を使って、辞書ドライブを RAM ディスクの D：にチェンジして、各種の日本語入力を行ってみましょう。その実行例を図 3.12 に示します。

```
A>COPY VJEB.DIC D:⏎ ……………辞書ファイルをRAMディスク上にコピーする
        1 個のファイルをコピーしました.

A>DIR D:⏎ ……………RAMディスク上のファイルを確認する

 ドライブ D: のディスクのボリュームラベルは M E L-DISK
 ディレクトリは D:¥                          ↑自動的にこのようなボリュームラベル
                                             が付けられている
VJEB     DIC   488448  89-08-09   21:34 ……………辞書ファイルがコピーされている
        1 個のファイルがあります.
  1439744 バイトが使用可能です.
    (VJE-βの機能で、辞書ドライブをドライブD：に設定しておく。CTRL+f·10→f·2の後、ドライブ名「D」を指定する)
A>おうよう ) XFER   各種の変換を行ってみる
  応用

かな カナ 全角 半角 ｺｰﾄﾞ  10確定  ^3ﾛｰﾏ字 ^10その他                Rかな全J*
```

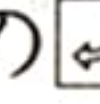



図 3.12　RAM ディスク上の辞書ファイルを使って日本語入力を行う

当然のことですが、非常に速くスムーズに各種の変換が行われることが実感できます。辞書ドライブを、フロッピーディスクのドライブ A：にチェンジして、その違いを比べてみると、この変換速度の改善は劇的と言えるでしょう。

ではもう一例、簡単なもので RAM ディスクの効果を実験してみましょう。まず、ドライブ B：に、ソートコマンドの「SORT.EXE」が含まれている MS-DOS のシステムディスクなど、適当なディスクをセットします。そしてカレントドライブを、フロッピーディスクの A：や、RAM ディスクの D：にチェンジして、ドライブ B：に対して DIR コマンドを実行し、パイプを使ってその表示出力のファイル名を ABC 順にソートしてみます。パイプによる SORT コマンドの実行には、その内部的な作業に、カレントドライブ上で数回のディスクの読み／書きが行われるため、その処理時間を比較しようというわけです。このコマンドを実行する際の⏎の入力から、ソート結果の表示が始まるまでの時間を両者で比較します。

まず、カレントドライブが通常のフロッピーディスク（A：）の場合の実行例を次ページの図 3.13 に示します。

```
A>DIR B: | B:SORT⏎           ……5インチ2HDフロッピーディスク上での実行
                              (SORTコマンドはドライブB：にあり、パスが設定されている)

       24 個のファイルがあります.   ……コマンド実行後(⏎後)、表示が開始されるまで約4秒
   376832 バイトが使用可能です.
 ディレクトリは B:¥
  ドライブ B: のディスクのボリュームラベルはありません.
CONFIG   1          95  88-07-13    0:00   このディスクのCONFIG.SYSファイル
CONFIG   2          73  88-07-13    0:00   には、「BUFFERS=20」の指定あり
CONFIG   3          41  88-07-13    0:00
COPY2    COM      3344  88-07-13    0:00
～～～～～～～～～～～～～～～～～～～～～～～～～～～～
PRINT    EXE      8856  88-07-13    0:00
RECOVER  EXE      5073  88-07-13    0:00
SHARE    EXE      8528  88-07-13    0:00
SORT     EXE      2138  88-07-13    0:00
SUBST    EXE     10728  88-07-13    0:00

A>
```

図 3.13　通常のフロッピーディスク上でのソート付き DIR コマンドの実行

では、これと同じことをカレントドライブをRAMディスク(D：)にチェンジして行ってみましょう。その実行例を示します。

```
A>D:⏎    ……カレントドライブをD：(RAMディスク)に変更

D>DIR B: | B:SORT⏎    ……ドライブD：上で、ドライブA：上の先に実行したディスクの
                        ディレクトリをソートして表示する

       24 個のファイルがあります.   ……表示が開始されるまで約1秒。フロッピー
   376832 バイトが使用可能です.         ディスクでは約4秒
 ディレクトリは B:¥
  ドライブ B: のディスクのボリュームラベルはありません.
CONFIG   1          95  88-07-13    0:00
CONFIG   2          73  88-07-13    0:00   (条件は図3.13の実行例と同じ)
CONFIG   3          41  88-07-13    0:00
COPY2    COM      3344  88-07-13    0:00
～～～～～～～～～～～～～～～～～～～～～～～～～～～～
PRINT    EXE      8856  88-07-13    0:00
RECOVER  EXE      5073  88-07-13    0:00
SHARE    EXE      8528  88-07-13    0:00
SORT     EXE      2138  88-07-13    0:00
SUBST    EXE     10728  88-07-13    0:00

D>
```

図 3.14　RAM ディスク上でのソート付き DIR コマンドの実行

この実行例によっても、RAM ディスクの高速性を実感できたことと思います。SORT コマンドの内部処理にともなう、カレントドライブへの読み／書きの処理速度が、フロッピーディスクと RAM ディスクでこの程度違うわけです。3.1 章で述べた CONFIG.SYS ファイルの「BUFFERS」指定の数を多く取っていないシステムでは、その差がとくに顕著に表れるでしょう。

■ RAM ディスク使用上の注意

RAM ディスク上で実行する場合に限らず、実行するソフトウェアによっては、実行が終了して、MS-DOS のプロンプト（A>、B>など）にもどる時点で、コマンドプロセッサの「COMMAND.COM」を再ロードするものがあります。よく経験する例は、ディスクをまるごとコピーするプログラムの「DISKCOPY.EXE(.COM)」などもその1つです。DISKCOPY コマンドで、MS-DOS システムが含まれていない（COMMAND.COM がない）ディスクをコピーしていて、コピーが終了して MS-DOS にもどろうとした場合、次のようなエラーメッセージが表示されることを経験したことがあるでしょう。

```
A>DISKCOPY A: B:⏎    ……ドライブA：上のディスク全体をドライブB：にそっくりコピーする
DISKCOPY version 4.00

ディスクのコピーを行います

送り側ディスクをドライブ A: に挿入してください   ……COMMAND.COMが含まれて
受け側ディスクをドライブ B: に挿入してください       いないディスクをセットした
準備ができたらどれかのキーを押してください⏎

照合中      残り   0 トラックです
0 10 20 30 40 50 60 70 80 90 100
████████████████████████████████(%)

コピーは終了しました
もう一度実行しますか(Y/N)N   ……コピー終了。MS-DOSにもどる

¥COMMAND.COM の入ったディスクをカレントドライブに差し込み，  ドライブA：にセットしたディス
どれかキーを押してください．                                  クにCOMMAND.COMファ
                                                              イルが存在せず、MS-DOSの
                                                              プロンプトにもどれない
```

図 3.15 MS-DOS システムの含まれていないディスクでの COMMAND.COM の再ロード

この場合、フロッピーディスクであれば、エラーメッセージで警告されてからでも、システムディスクに交換すれば済むことですが、これがディスクの交換などできない RAM ディスクであれば、場合によっては悲劇となります。RAM ディスクでは、リセットボタンを押して MS-DOS を再起動するより手はなく、その場合、古いタイプの RAM ディスクでは、リセットによってメモリの内容が消えてしまい、取り返しがつかない事態になる場合もあります。

ただし最近のメモリボードの多くは、「ウォームブート機能」(リセットボタンにより MS-DOS を再起動しても、RAM ディスクのメモリ内容を保護する機能)を備えていますので、リセットによって RAM ディスクのメモリ内容が失われることはありません。ただし、プロテクト増設メモリ上で RAM ディスクを実現している場合は、リセット時や電源 ON 時のマシン自身によるメモリチェック機能により、メモリ内容がすべてクリアされてしまいますので注意してください。

いずれにしても、このようなトラブルを避けるには3通りの方法があります。

---

①あらかじめ RAM ディスク上に、COMMAND.COM をコピーしておく
② COMMAND.COM の再ロードが、常にドライブ A：から行われるように、その指定を CONFIG.SYS ファイルに登録しておく
③環境変数 COMSPEC にドライブ A：の COMMAND.COM を指定しておく
(ただしいずれの場合も、COMMAND.COM のバージョンは、同一のものでなければならない)

---

①の方法は、MS-DOS を起動した時点で、COPY コマンドを使って、システムディスクから RAM ディスクに COMMAND.COM をコピーしておけばよいわけです。あるいは、RAM ディスクに設定されるドライブ名が常に決まっているならば(つまり、ディスク装置に関するハードウェア環境が常に決まっているならば)、自動スタート・バッチファイル「AUTOEXEC.BAT」に、そのコピーコマンドを登録しておき、MS-DOS の起動時に自動的に実行させるのもよいでしょう。その自動スタート・バッチファイルを作成する実行例を次に示します。ここでは、ドライブ D：が常に RAM ディスクに設定されることを前提にしています。

```
A>COPY CON AUTOEXEC.BAT ↵ ………COPYコマンドを使って、自動スタートのバッチファイルを作成する
COPY COMMAND.COM D:↵ ………カレントドライブ(A：)上のCOMMAND.COMを
                             ドライブD：(RAMディスク)上にコピーするコマンド
^Z↵ ………CTRL+Z ↵
        1 個のファイルをコピーしました.

A>TYPE AUTOEXEC.BAT↵ ………作成されたファイルの内容をタイプアウトして確認する
COPY COMMAND.COM D: ………MS-DOSの起動時に、このコマンドが実行される

A>
```

図 3.16　起動時に RAM ディスクへ COMMAND.COM をコピーする自動スタート・バッチファイルの作成

この自動スタート・バッチファイルにより、MS-DOS が起動した時点で、ドライブ D：の RAM ディスクに COMMAND.COM が自動的にコピーされます。

もしすでに自動スタート・バッチファイルが存在している場合は、エディタやワープロを使って、この COPY コマンドを適当な位置に追加します。あるいは COPY コマンドを使って、ファイル全体を書き直してもよいでしょう。

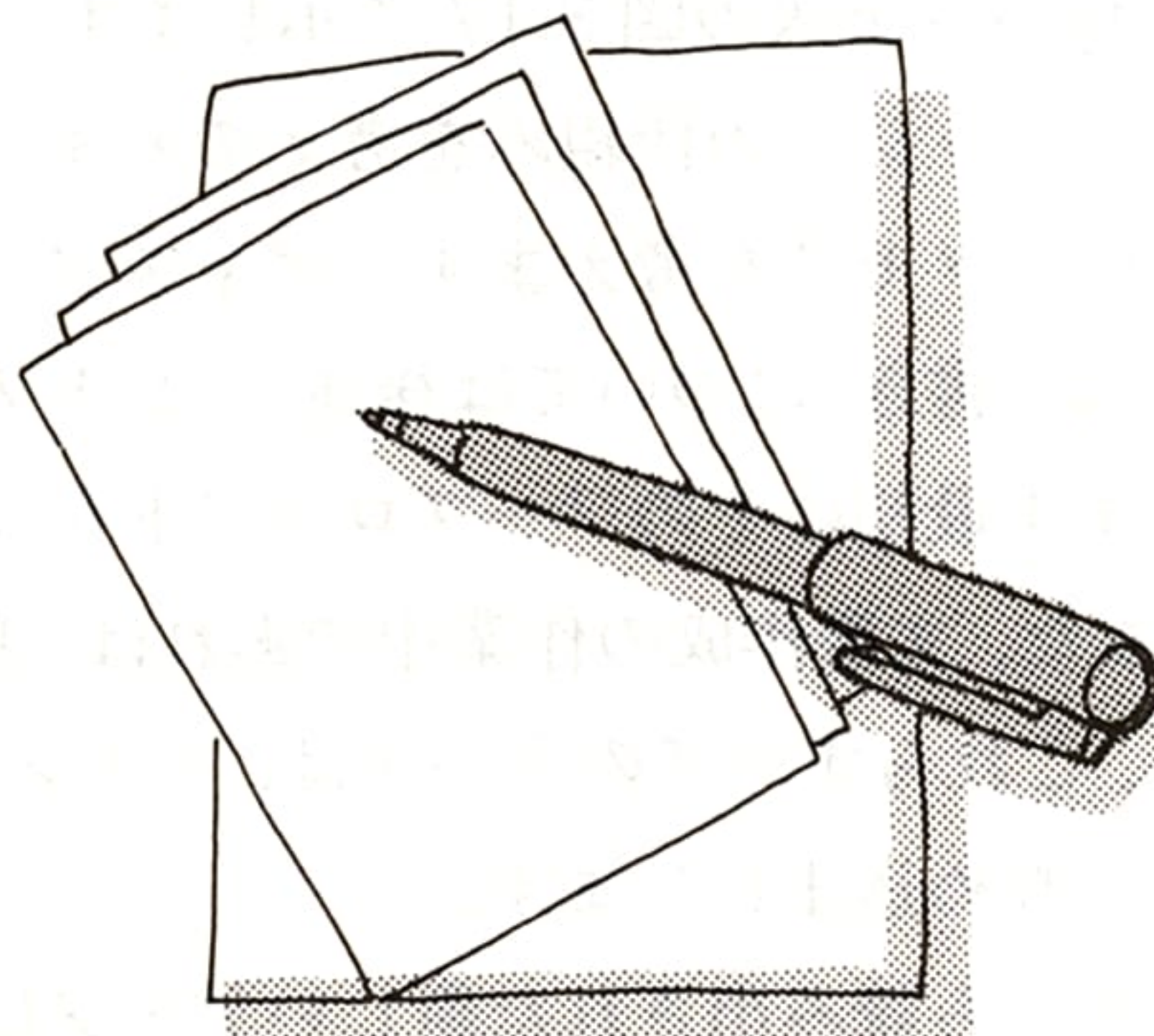
②の方法は、次の１行を CONFIG.SYS ファイルに追加します。

```
SHELL＝A：¥COMMAND.COM  A：¥
```

これは、SHELL コマンドによって、COMMAND.COM の再ロードを、常に「A：¥」上の「COMMAND.COM」で行うように設定するものです。なお、この SHELL コマンドによって設定されたドライブ名やパス名は、結局、次の③の環境変数「COMSPEC」に登録されることになります。

③の方法は、環境変数「COMSPEC」に COMMAND.COM のあるドライブを指定します。COMSPEC は、MS-DOS システムが COMMAND.COM を再ロードする際に参照する特殊な環境変数で、たとえば MS-DOS のプロンプトから次のように設定します。あるいは①と同様に、自動スタート・バッチファイルに登録しておけばよいでしょう。

```
A>SET  COMSPEC＝A：¥COMMAND.COM
```

このコマンドの実行後、COMMAND.COM の再ロードは常にドライブ A：から行われます。この方法は、②の方法で指定した COMMAND.COM の再ロード先を、あとから別のドライブに変更するときに用いることができます。

3.5 ディスクキャッシュによる高速化

RAM ディスクは、もっとも高速の「ディスク装置」ではありますが、電源を切るとメモリの内容が消滅するという本質的な〝特長〟を持っています。いわゆる**揮発性メモリ**と呼ばれているものです。RAM ディスクのメモリ内容を、コンピュータ本体の電源が切られても保持するためには、そのメモリボードに対して別系統から電源を供給したり、バッテリーでバックアップしたり、また消費電力がきわめて少ないメモリ素子でメモリボードを構成するなどの方法が取られています。

しかし、いくらこれらの方法を取ってみても、本質的にはやはり揮発性メモリであることには変わりなく、フロッピーディスクやハードディスクなどの**不揮発性メモリ**に比べると、「記憶内容の保持」に関する信頼性は比較になりません。私たちは、RAM ディスクの高速性を利用しますが、それはあくまで、一時の作業用としての利用に限られ、「主外部記憶装置」としては、やはり通常のディスク装置を使うほかにありません。その RAM ディスクの高速性を、なんとか通常のディスク装置で実現できないものか…。無理難題とも言えるその要求に、巧妙な仕掛けで応えたものが、本節で解説する「ディスクキャッシュ」という方法です。

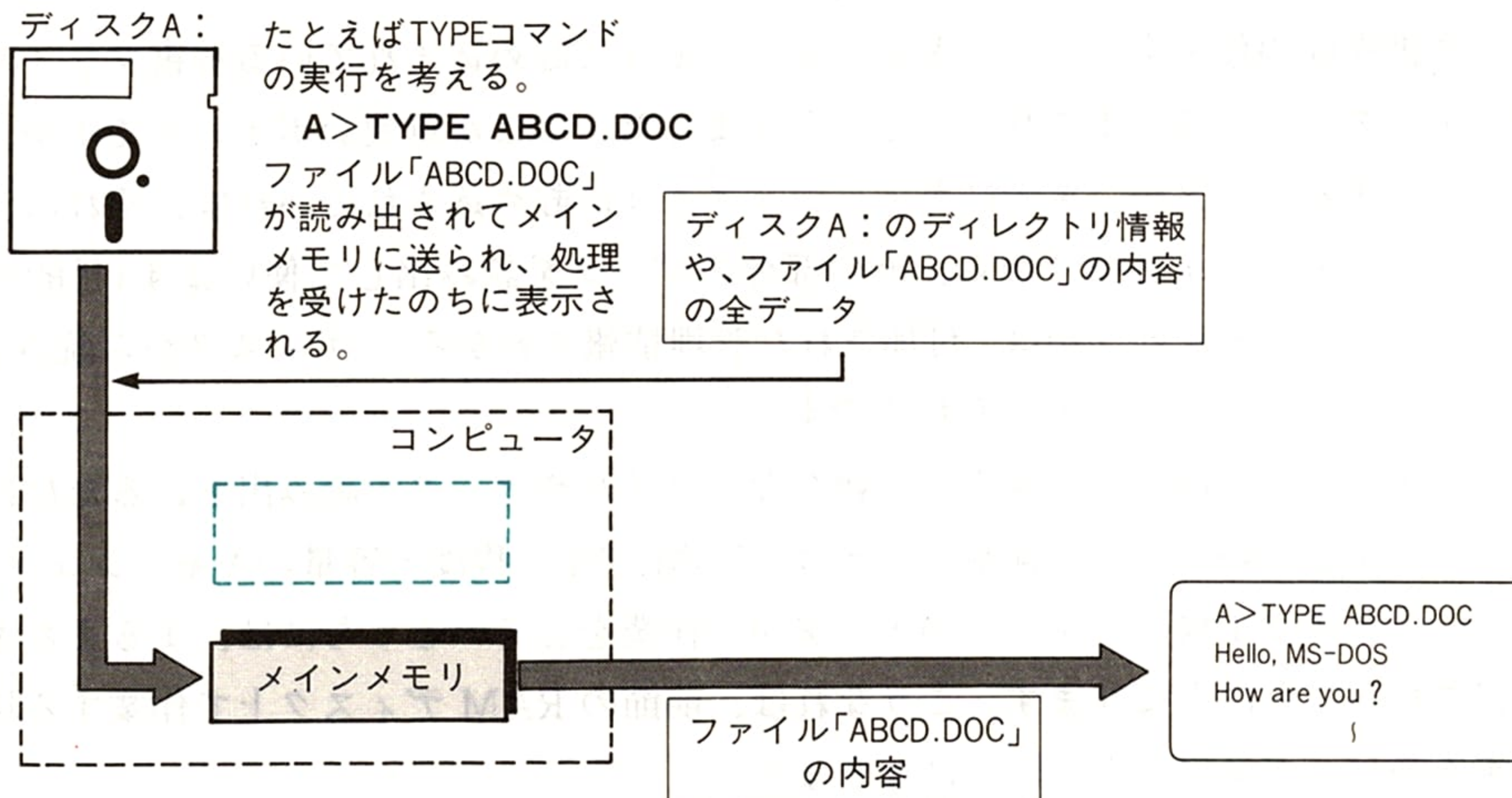
■ ディスクキャッシュの仕組み

ディスクキャッシュは、フロッピーディスクやハードディスクの「読み出し」(読み出しのみ)を、RAM ディスクと同等の速度まで高めることを可能にした、ディスクの読み出しデータの「バッファ機能」と考えることができます。本章の 3.1 で解説した「ディスクバッファ」と同じ原理と考えればよいでしょう。ディスクから読み出したデータをメモリ上に保管しておき、再び同じ部分の読み出しが発生した場合には、再度ディスクから読み出すことなく、メモリ上の該当データをそのまま使うという仕掛けです。

ただしディスクキャッシュの威力は、ディスクの読み出しに対して発揮されるだけであり、書き込みに対しては、まったく何の効果もありません(厳密に測定すると、ほんの僅か遅くなるはず)。そのディスクキャッシュの仕組みを次の**図 3.17** で示します。

この図からディスクキャッシュの仕組みを考えてみましょう。まず、増設した RAM ボード上に、「キャッシュメモリ」というメモリを考えます。ディスクの読み／書きには、すべてこのキャッシュメモリの参照が行われます(「書き」については後述)。たとえば、ワープロソフトにおけるディスクの読み／書きを考えると、まず最初にそのワープロソフトを立ち上げる際、ワープロのプログラムファイルが読み出されます。また、文書作成の作業中であれば、辞書ファイルが盛んに読み出されます。ディスクから読み出された、これらすべてのデータは、メインメモリに取り込まれて処理されると同時に、キャッシュメモリにも〝貯め込まれ〟ます。

このように、ディスクから読み出された新しいデータは、すべて、キャッシュメモリにどんどん貯

■ディスクキャッシュが設定されていない場合

■ディスクキャッシュが設定されている場合

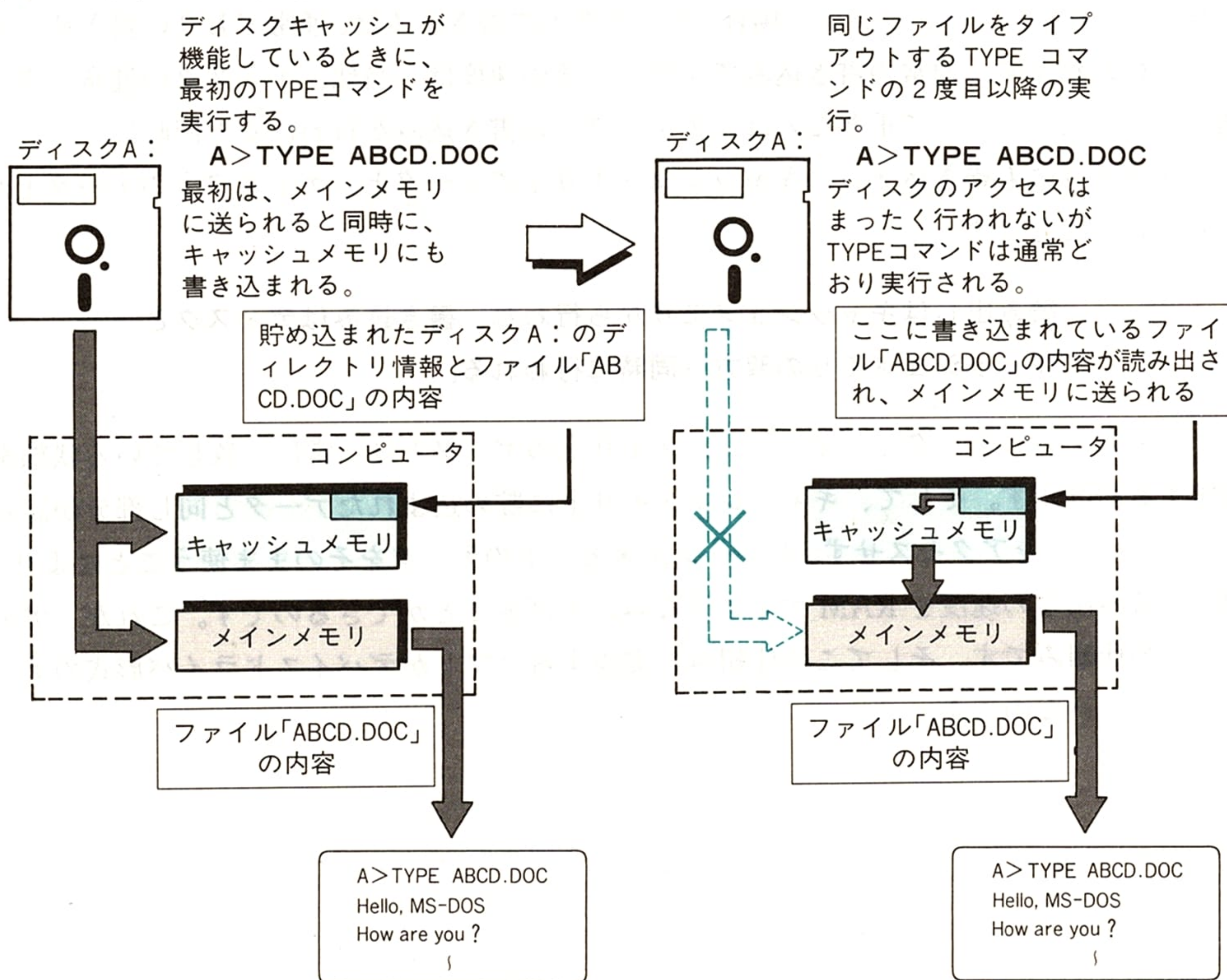



図3.17　ディスクキャッシュの仕組み

め込まれていきます。それも、「どのディスクのどこの部分のデータ」であるかを判別するための管理情報を付加し、きちんと管理されて貯め込まれるのです。

さて、文書作成の作業中、すでにキャッシュメモリに貯め込まれている辞書ファイルのデータと同じ部分がディスクから読み出されることになりました。ここが重要なポイントですが、ディスクから読みだそうとするデータが、すでにキャッシュメモリに貯め込まれていれば、それはディスクから読み出さずに、キャッシュメモリから、その部分のデータを読み出して使います(目的の部分がキャッシュメモリに存在するかどうかは、付加された管理情報でわかる)。ディスクから読み出すより、はるかに速く目的のデータを読み出せるわけです。

キャッシュメモリには、ディスクから新たなファイルやデータが読み出されるたびに、それらがどんどん貯め込まれていきます。極端にいえば、1.5Mバイト程度の容量のキャッシュメモリであれば、フロッピーディスク1枚分のすべてのデータが、作業をしているうちには、まるまるキャッシュメモリに収容されることにもなります。こうなれば、前節のRAMディスク上で作業する場合と、読み出しに関するスピードはほとんど同じになります。

しかし、RAMディスクとは仕組みが根本的に異なり、とくにディスクへの書き込みに関しては決定的に異なります。ディスクキャッシュの場合、ディスクへの書き込みは、通常どおりの書き込みがディスクに対して行われます。通常の書き込みですので、その速度は、当然、ディスクの通常の書き込み速度と同じです。ただしここで重要なのは、ディスクへの書き込みが行われると同時に、そのデータがキャッシュメモリにも書き込まれ、キャッシュメモリ上のデータと、ディスク上のデータとが常に同じになるということです。

読み出しはキャッシュメモリから行われ、書き込みはディスクとキャッシュメモリの双方へ同時に行われる。

つまり、ディスク上のデータと、キャッシュメモリ上のデータとは、常に一致している状態を保つように動作するわけです。そして、キャッシュメモリ上に貯め込まれたデータと同じ部分が読み出されるときは、ディスクをアクセスせず、キャッシュメモリ上のデータをそのまま使うことにより、ディスクに関する読み出しの速度をRAMディスクなみに上げることができるのです。これが「ディスクキャッシュ」の仕組みです。そしてこの仕組みを実現するソフトがデバイスドライバ形式のディスクキャッシュドライバなのです。

ディスクA：

たとえば、図3.17で登場したファイル「ABCD.DOC」がエディタやワープロによる変更を受けたとする。
ディスクへ書きもどす際に、キャッシュメモリ上の同ファイルのデータや、ディスクA：に関するディレクトリ情報も同時に書き直される。つまり、キャッシュ上のデータと、ディスク上のデータは同じになる。また新規作成ファイルなどのキャッシュメモリ上に存在しないデータの場合も、キャッシュメモリとディスク上の両方に書き込まれる。

ディスクに書きもどされるファイル「ABCD.DOC」、およびディスクA：のディレクトリ情報

コンピュータ

キャッシュメモリ

書き直されるファイル「ABCD.DOC」、およびディスクA：のディレクトリ情報

メインメモリ

A＞EDLIN ABCD.DOC
Hollo MS-DOS
See you Again

ファイル「ABCD.DOC」の編集作業をして、ディスクに書きもどす

図3.18　書き込みは、ディスクとキャッシュメモリの双方へ行われる

■ MELWARE によるディスクキャッシュの設定

本章の3.3で述べたように、ディスクキャッシュを実現するためには、次の2つのデバイスドライバのプログラムファイルが必要です。

- MELEMM.SYS　………EMMドライバ（本章3.6で解説）
- MELCACHE.SYS ………ディスクキャッシュドライバ

このデバイスドライバを動作させるためには、CONFIG.SYSファイルへ次の2行を登録しておく必要があります。

例：
```
DEVICE = MELEMM.SYS  /U0
DEVICE = MELCACHE.SYS 1920
```

RAMディスクの節でも述べたように、これらのデバイスドライバは、任意のディレクトリ上に置くことが可能ですが、その場合は、それぞれのプログラムファイルにパス名を付ける必要があります

(「DEVICE＝¥RAM¥MELEMM.SYS /U0」というように)。なお、「DEVICE＝MELEMM.SYS /U0」の行は、EMS を設定する行ですが、ディスクキャッシュを設定する前提として、EMS を使わない場合にも必要です。RAM ディスクの節を参照してください。

次の行、「DEVICE＝MELCACHE.SYS　1920」の行は、ディスクキャッシュを設定する行です。「1920」の部分はオプションパラメータの一例ですが、その一般的な書式を次に示します。

DEVICE ＝ MELCACHE.SYS (メモリ容量)／(ドライブ指定) (S) (L)

- 「メモリ容量」は確保するキャッシュメモリの容量(K バイト単位で指定。省略した場合は増設ボードの全メモリ)
- 「ドライブ指定」はディスクキャッシュを行うディスクドライブの指定
  Hn…ハードディスク
  Fn…1M バイトタイプのフロッピーディスク
  Dn…640K バイトのフロッピーディスク
  n はドライブ番号で Hn は 1～2、Fn・Dn は 1～4。省略した場合は全ドライブ
- 「S」を付けると、メモリのサムチェック(キャッシュメモリの読み出し時のデータチェック)を行う(省略した場合はサムチェクなし)
- 「L」を付けるとフロッピーディスクドライブのドアを開けたときのメモリクリア(後述)を行わない

たとえば、「DEVICE＝MELCACHE.SYS　1920　/H1　F1　S」は、バンクメモリの 1920K バイトをキャッシュメモリ領域として確保し、ハードディスクの No.1、フロッピーディスクの No.1 に対してディスクキャッシュを行い、データのサムチェックを行うという設定のディスクキャッシュを実現する指示となります(1920 という値は、2M バイトの RAM ボードを〝バンク切り替え方式〟で使用する場合のもので、2M、つまり 2048K バイトのなかの 128K バイトは、メインメモリとして使われるため、キャッシュメモリはその 128K バイト分減少します)。

また、メニュー方式により、RAM ディスク／ディスクキャッシュ／EMS の各設定を簡単に行うことができる「簡単設定プログラム」が用意されていますが、これについては本章の最後にまとめて示します。

では、1.4 章で使った日本語入力システムの VJE を組み込んだシステムディスク上に、ディスクキャッシュを設定する一連の実行例を示しましょう。MELWARE のオリジナルディスク(またはそのバックアップコピー)をドライブ A：に、VJE を組み込んだシステムディスクをドライブ B：にセットして作業を始めます。

```
A>COPY MELEMM.SYS B:⏎ ……………EMMドライバをコピーする
         1 個のファイルをコピーしました.

A>COPY MELCACHE.SYS B:⏎ ……………ディスクキャッシュドライバをコピーする
         1 個のファイルをコピーしました.

A>DIR B:⏎ ……………ディスクの内容を確認する

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

COMMAND  COM     24931  88-10-16   16:10
VJEB     SYS     76913  89-01-31    2:00
VJEB     DRV      2763  88-10-10    2:00   1.4章で述べたVJE-β用の各種ファイル
VJEB     DIC    488448  89-01-08    2:00
SETVJE   EXE     15394  88-11-01    2:00
CONFIG   SYS        84  89-09-07   21:57
MELEMM   SYS     16416  89-02-21   11:43   MELWAREのオリジナルディスクから、
MELCACHE SYS      9253  89-01-18   15:02   ディスクキャッシュを設定するのに必要な2つ
                                           のデバイスドライバがコピーされた
         8 個のファイルがあります.
   515072 バイトが使用可能です.

A>
         ⋮
  途中省略(CONFIG.SYSファイルに、下記のデバイスドライバを
  エディタやCOPYコマンドを使って登録する)
         ⋮

B>TYPE CONFIG.SYS⏎ ……………ディスクキャッシュを設定するためのデバイスドライバを
                            組み込んだCONFIG.SYSをタイプアウトする
FILES=10
BUFFERS=20
DEVICE = MELEMM.SYS /U0 ……… オプション「/U0」は、EMMドライバをバンク管理用のマネージャとしてのみ
                             使用することを示す
DEVICE = MELCACHE.SYS 1920 ……… ディスクキャッシュドライバの本体。ここでは全ドライブを対象に
                                 1920Kバイトのディスクキャッシュを設定する
DEVICE = VJEB.DRV /HR /M1 /T0 /L /J /K- /K, /K. /D4

B>
```

図3.19　VJEを組み込んだMS-DOSのシステムディスク上にディスクキャッシュを設定する

なお、CONFIG.SYSファイルの書き換え(2行追加)は、エディタやワープロを使って行いますが、COPYコマンドを使って、最初から全体を作り直してもかまいません。

以上のように設定した後、このシステムディスクでMS-DOSを再起動すれば、ディスクキャッシュが機能します。ではディスクキャッシュが設定されたシステムディスクをドライブA：にセットして、リセットボタンを押してみましょう。

■ディスクキャッシュを使ったユーザープログラムの実行

さて、ディスクキャッシュが設定されたシステムディスクを起動すると、MS-DOSのオープニングメッセージとともに、ディスクキャッシュなどのデバイスドライバがシステムに組み込まれ、その際の各種のメッセージが表示されます。

リセットボタンを押す

```
NEC PC-9800 Series Personal Computer

ﾏｲｸﾛｿﾌﾄ MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30
Copyright 1981,88 Microsoft Corp. / NEC Corporation

    》バンクメモリーマネジャー《      [Ver2.21] Jan.1989 MELCO Inc. Japan

キャッシュドライバー《ＭＥＬ＿ＣＡＣＨＥ》を組み込みました。
[Ver 2.10] Copyright 1989  MELCO.Inc Japan
全てのドライブがキャッシング対象になります。
[ １９２０ＫＢ] を確保しました。

日本語入力フロント・プロセッサ ＶＪＥ－β 第２.００版
Copyright (C) 1986-88 VACS Corp. / ASCII Corp.

現在の日付は 1989-09-06 (水) です.
日付を入力してください:
現在の時刻は 23:06:30.00 です.
時刻を入力してください:

A>
```

MS-DOSのオープニングメッセージ

EMMドライバ/ディスクキャッシュドライバの組み込みメッセージ。すべてのドライブを対象に1920Kバイトのディスクキャッシュが設定されている

VJE-βのFEPドライバ組み込みメッセージ

A>……………システムが完全に立ち上がった。全ドライブにおいてディスクキャッシュが行われる

図3.20　ディスクキャッシュが組み込まれたことを示すMS-DOS起動時のメッセージ

このメッセージには、すべてのディスクドライブがディスクキャッシュの対象であり(実際には、必要なドライブに限定する方がよい)、そのキャッシュメモリの容量は1920Kバイトであることなどが表示されています。つまり、接続されているドライブ全体で、容量1920Kバイトのキャッシュメモリを使うことができるわけです。

ディスクキャッシュを行うディスクと行わないディスクの指定は、CONFIG.SYSファイルへのデバイスドライバの登録時のオプションの設定で可能ですが、ディスクキャッシュが設定されたドライブに関しては、キャッシュメモリを使う優先順位や、ドライブの種類による差別はありません。多くのファイル、多くのデータを読み出したドライブが、多くのキャッシュメモリを占有することになります。

ではまず、果たしてディスクキャッシュが機能するかどうかを、簡単な実験で試してみましょう。ドライブB：に適当なテキストファイル形式の文書ファイルが存在するディスクをセットして、それをTYPEコマンドで読み出してみます。MS-DOSのシステムディスクやMELWAREのオリジナルディスクに含まれているドキュメントファイルを利用すればよいでしょう。ここではMELWAREのディスクに含まれるドキュメントファイル「MANUAL.DOC」をタイプアウトしましょう。

```
B>TYPE MANUAL.DOC ↵ ……………MELWAREのディスクにはいっている「MANUAL.DOC」をタイプアウトする

MANUAL.DOC *** CONTENTS *** ……………最初の読み込みなので、ディスクアクセスが行われて
                                    タイプアウトが実行される
このマニュアルは．．．

　上級ユーザーの方が、MELWAREディスクに含まれる機能をフル活用するための情
報について記述されています。　　　　　　　初心者の方は読み飛ばしていただいてかま
本書の内容を無断で複製することを禁止

B>
```

図3.21　ドキュメントファイル「MANUAL.DOC」をタイプアウトする。最初のアクセス

いつものように、何の変わりもなく「MANUAL.DOC」がタイプアウトされました。さて、ここからが実験です。ここで CTRL ＋ C をキー入力して、再度「MANUAL.DOC」をタイプアウトしてみましょう。CTRL ＋ C を入力する意味は、本章の3.1で解説したディスクバッファリングの機能を、ここでクリアするためです。ディスクキャッシュが機能していなければ、再度のTYPEコマンドで、必ず目的のディスクとそのファイルがアクセスされるはずです。では実行してみましょう。

```
B>^C ……………CTRL＋Cを入力して、ディスクバッファをクリアする（3.1章を参照）
B>TYPE MANUAL.DOC ↵ …………再度、前述のファイルをタイプアウトする

MANUAL.DOC *** CONTENTS *** ……………ディスクアクセスがまったく行われずにタイプアウトが実行される

このマニュアルは．．．

　上級ユーザーの方が、MELWAREディスクに含まれる機能をフル活用するための情
本書の内容を無断で複製することを禁止　　　　は読み飛ばしていただいてかま

B>
```

図3.22　同じドキュメントファイルを再度タイプアウトする。2度目のアクセス

目的のディスクは、まったくアクセスされることなく、TYPEコマンドの入力とほとんど同時に、ドキュメントファイルの内容の表示が開始されました。目的のディスクのディレクトリ情報も、ファイルの中身も、すでにキャッシュメモリに貯め込まれているため、ディスクをアクセスする必要がないのです。ディスクキャッシュは、正常に機能していることが確認できました。

さて、この一例からも推測できますが、ディスクキャッシュの使い方に関しては、それが機能していることを、ユーザーはまったく気にする必要はありません。前節のRAMディスクの場合は、「揮発性ディスク」であることから、通常のディスクとは違った注意や使い方なども必要でしたが、ディスクキャッシュの場合は、ディスク装置は「まったくの通常ディスク」として使えます。いつものように作業すれば、ディスクから読み出されたデータは、それがどのようなものであろうと、キャッシュメモリに貯め込まれ、貯め込まれたそのデータに関しては、2度目以降の読み出しは、キャッシュメモリ上のものがそのまま読み出されて処理されるのです。

ですから、それぞれのディスクの使い始め（つまりそれぞれのデータが最初に読み出されるころ）は、ディスクキャッシュの効果はまったくありません。ディスクの読み出しがともなう作業を進めていくうちに、めきめきとディスクキャッシュの効果が出てきます。

ワープロの「かな－漢字変換」（つまりディスク上の辞書ファイルの検索）にも、ほぼRAMディスクなみの威力を発揮します。しばらくワープロ作業をすることにより、辞書ファイルのかなりの部分が、キャッシュメモリに貯め込まれることになるからです。

また意識的に、あらかじめ目的のファイルのデータを、キャッシュメモリ上に貯め込んでおく積極的な方法もあります。目的のファイルに対してCOPYコマンドを実行すればよいのです。たとえばワープロ一太郎の辞書ファイルを、あらかじめキャッシュメモリに貯め込んでおく場合は、次のようにCOPYコマンドを実行します。

```
A>COPY ATOK.DIC NUL ⏎
```

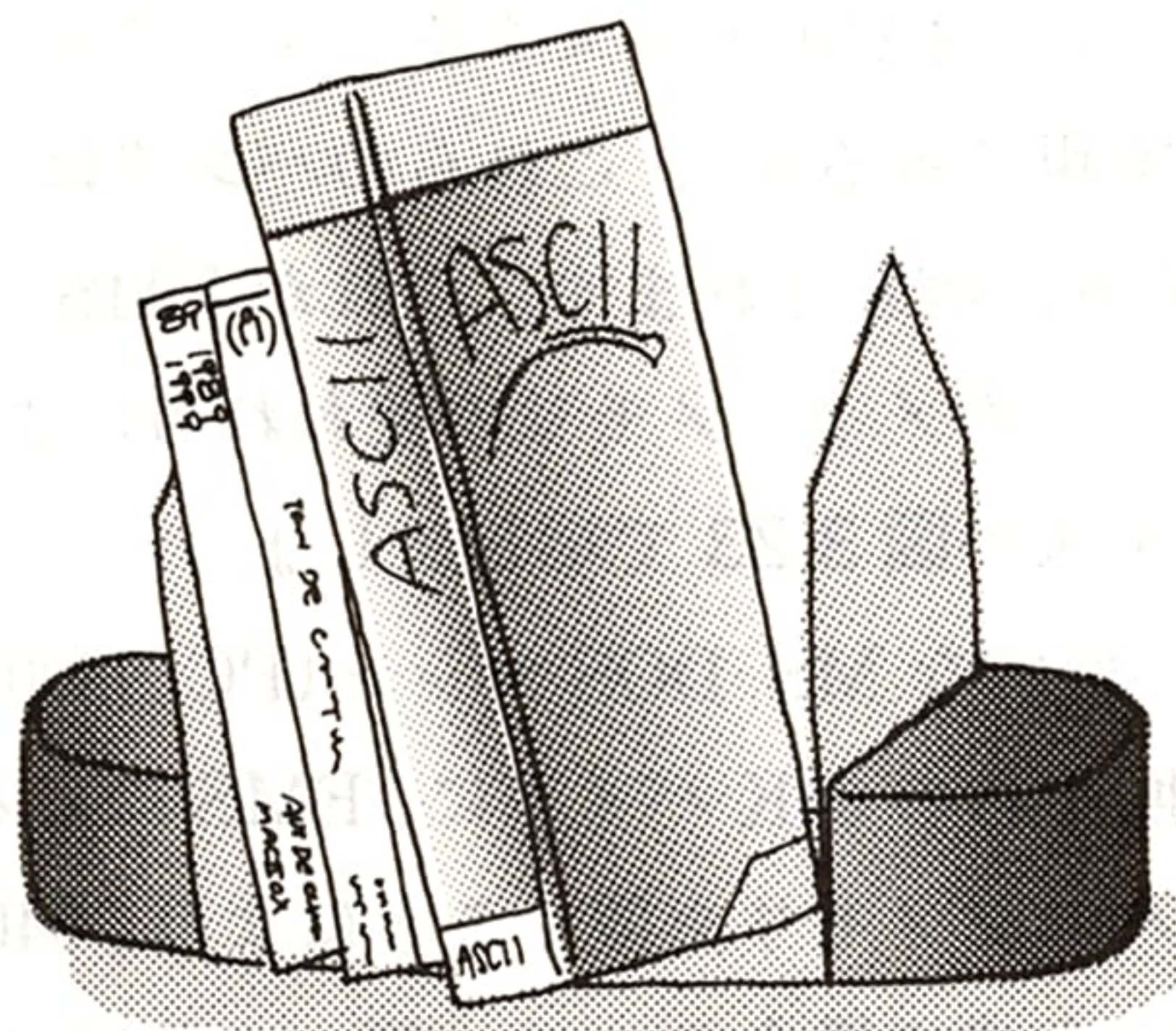
このCOPYコマンドは、ファイル「ATOK.DIC」をNUL（ヌル）にコピーします。NULとは「無」（ダミー）のことですから、読み出された「ATOK.DIC」のデータのコピー先は「無」であり、COPYコマンドは実行されますが、そのデータはただ捨てられるだけでCOPYコマンドの実行は終わります。しかしアクセスされた辞書ファイル「ATOK.DIC」のデータは、すべてキャッシュメモリに貯め込まれますので、かな－漢字変換は、最初からほとんどRAMディスクの快適さで行うことができるわけです。

■ ディスクキャッシュ使用上の注意

ディスクキャッシュが機能するための大前提は、キャッシュメモリの内容と、実際のディスクドライブの内容とが同じものであることが保証されていることです。これは本章3.1で解説したディスクバッファリングの前提とまったく同じです。ですから〝知らない間に〟ディスクが入れ替えられては致命的なダメージを受けることになります(通常のハードディスクは、ディスクの入れ替えができないので、対象はフロッピーディスクですが)。

そこで、もしフロッピーディスクの入れ替えが行われた場合には ―― 具体的には、ディスクドライブのドアが開閉された場合 ―― キャッシュメモリの、そのドライブに関するデータは、すべて破棄されます。ほかのドライブに関するデータはそのままの状態で、影響はありません。このこともディスクバッファリングと似ていますが、ディスクバッファリングの場合は、すべてのディスクのデータがクリアされる点が異なります。ただし、ディスクキャッシュを実現するソフトによっては(ここではMELWAREですが)、どのディスクドライブのフロッピーディスクを交換しても、キャッシュメモリ全体が(つまり、すべてのディスクのデータが)破棄されるものもありますので注意してください。

したがって、頻繁にフロッピーディスクの交換をするような使い方や、そのようなアプリケーションプログラムの実行には、そのドライブに関するディスクキャッシュの効果はたいして期待できません。そのようなドライブには、ディスクキャッシュを設定しない方がよいでしょう。当然ながら、ディスクの交換がないハードディスクには非常に有効です。

また増設RAMボードが、バンクメモリであっても、プロテクト増設メモリであっても、いずれの場合も、マシンをリセットすればキャッシュメモリは全面的にクリアされてしまいますので、リセットボタンの操作は、その点を考慮にいれて行ってください。

3.6 EMSによるメモリの拡張

「EMS」とは、〝Expanded Memory Specification〟(拡張メモリ仕様)のことです。EMSによって実現される機能自体は、けっして新しいことでも特別なことでもありません。EMSは、それまで統一されていなかった「バンク切り替え方式」による拡張メモリ(本章の3.2で解説した)の仕様を、ロータス社、インテル社、マイクロソフト社の3社(それぞれの頭文字を取って、〝LIM〟と呼ばれる)が統一したものにすぎないのです。

EMSは、ハードウェア的には、ただのバンク切り替え方式の拡張メモリです。すでに述べたように、一般的なMS-DOSマシンにおける、ユーザーが使えるメインメモリの容量は最大640Kバイトであり、プログラムの容量や、扱うデータがますます大きくなりつつある最近のビジネスソフトなどの実行に、この制約は大きな問題となってきました。

以前より、各ボードメーカーなどが、それぞれの仕様によるバンク切り替え方式のメモリ拡張RAMボードを販売しており、それを私たちは、RAMディスクやディスクキャッシュなどに使っていましたが、そのバンクメモリの仕様を統一して、アプリケーションプログラムの実行にも広く利用できるようにしたものが「EMS」なのです。ただし、アプリケーションプログラムにおいてEMSによる拡張メモリを利用するには、あらかじめ、そのように作られたアプリケーションプログラムでなければなりません。

■ EMSの仕組み

EMSによる拡張メモリは、バンクメモリですので、その基本は本章3.2の「メモリの拡張」で述べたことと同じです。ですから、バンク切り替え方式などの基礎知識は、すでにあるものとしてEMSを解説しましょう。

従来のバンクメモリと異なる主な点は、バンクメモリが位置するアドレス(メモリの番地)と、バンクの内部構造です。本章の3.4や3.5では、現在一般的に使われている増設RAMボードを、バンクメモリとして動作させ、RAMディスクやディスクキャッシュを実現しましたが(プロテクトモード増設メモリとしても実現できるが)、これらのバンクは、640Kバイトの「ユーザーエリア内」の最上位の128Kバイトの位置に設定されます。これがEMSでは「ユーザーエリア外」に追い出されています。その位置は各マシンのメインメモリの使われ方によって、ユーザー側で決められるようになっていますが、その状態を次の**図3.23**で示しましょう。

このように、一般的なMS-DOSマシン(PC-9800シリーズなど)では、普通、メインメモリのアドレス「0C0000H番地」からの64Kバイトに、EMSによるバンクメモリが設定されます。このように、ユーザーエリア外にバンクメモリを置くことにより、640Kバイトがまるまる使用できるようになり、従来のバンク切り替え方式の場合の、ユーザーエリアにおけるバンク部128Kバイト分の使用上の制約が

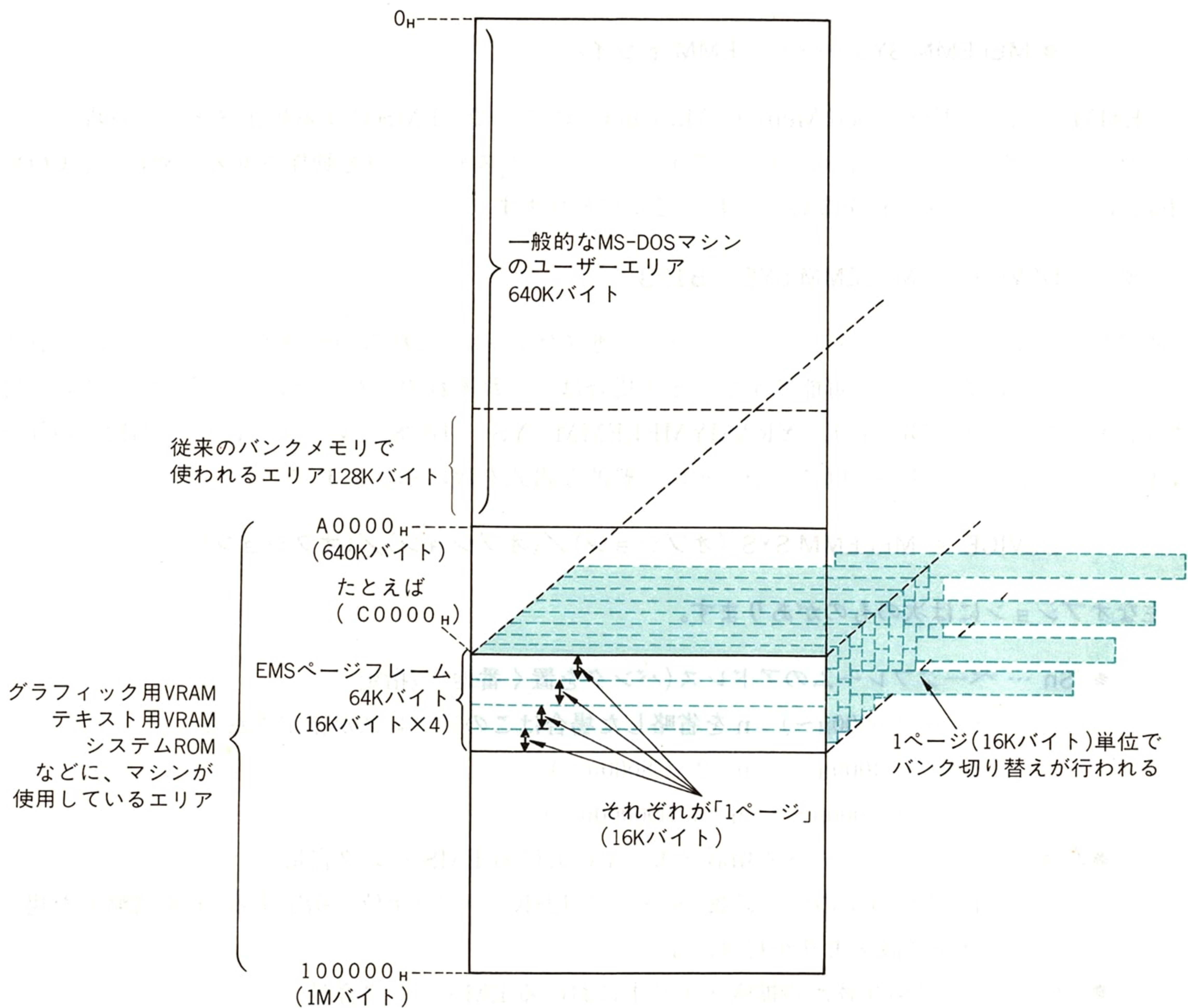



図 3.23　EMS によるバンクメモリの仕組み

なくなりました。

また、EMSによるバンクは、16Kバイトを単位(1ページ)とする4ページ、合計64Kバイトのエリア(これを**ページフレーム**と呼ぶ)を、1ページ単位(16Kバイト単位)でバンク切り替えができるように構成されています。これにより、従来の128Kバイト単位でのバンク切り替えよりも、柔軟で細かい応用が可能になります。

■ MELWARE による EMS の設定

EMS を実現するためには、次のデバイスドライバのプログラムファイルが必要です。

● MELEMM.SYS ……… EMM ドライバ

「EMM」とは、〝Expanded Memory Manager〞のことで、EMS による拡張メモリを管理するプログラム(デバイスドライバ形式)のことです。このデバイスドライバを動作させるためには、CONFIG.SYS ファイルへ次の行を登録しておく必要があります。

例： DEVICE = MELEMM.SYS /B128

RAM ディスクやディスクキャッシュの節でも述べたように、これらのデバイスドライバは、任意のディレクトリ上に置くことが可能ですが、その場合は、それぞれのプログラムファイルにパス名を付ける必要があります(「DEVICE=¥RAM¥MELEMM.SYS /B128」というように)。「/B128」の部分はオプションパラメータの一例ですが、その一般的な書式を次に示します。

DEVICE = MELEMM.SYS (オプション)／(オプション)／(オプション)…

主なオプションには次のものがあります。

●/Sn … ページフレームのアドレス(バンクを置く番地)の指定
n=0 (C0000H～) n を省略した場合はこのアドレスが指定される
n=1 (C4000H～) n=2 (C8000H～)
n=3 (CC000H～) n=4 (D0000H～)

●/Vn … プロテクトモードの増設メモリ上における EMS バンク容量
n=4 桁以下のページ数。8 ページ(128K バイト)単位で指定する。n を省略した場合は全増設メモリが対象になる

●/Bn … バンク切り替えの拡張メモリ上における EMS バンク容量
n=4 桁以下のページ数。8 ページ(128K バイト)単位で指定する。n を省略した場合は全拡張メモリが対象になる

●/U0 … デバイスドライバ「MELEMM.SYS」の、バンクマネージャのみを組み込む
(EMS そのものは利用せず RAM ディスクやディスクキャッシュを利用する場合)

たとえば、「DEVICE=MELEMM.SYS /B120 /S4」は、バンクメモリの 1920K バイト(16×120=1920)を EMS バンクメモリとして確保し、そのバンクをメインメモリのアドレス D0000H からの番地に設定する指示となります(ただし、EMS ドライバが 16K バイトとるので実際には 119 ページ分 1904 バイトが使用可能)。1920 という値は、ここで使っている HCE-2000 のような、ハードウェア対応の EMS ボードではない一般的な RAM ボードを〝バンク切り替え方式〞で使用する場合のもの

で、2M、つまり 2048K バイトのなかの 128K バイトは、メインメモリとして使われるため、EMS メモリは、その 128K バイト分減少します。

では、1.4 章で使った日本語入力システムの VJE を組み込んだシステムディスク上に、EMS バンクメモリを設定する一連の実行例を示しましょう。ここではバンク切り替えの RAM ボード上に(つまりプロテクト増設メモリ上ではなく)EMS を実現します。MELWARE のオリジナルディスク(またはそのバックアップコピー)をドライブ A：に、VJE を組み込んだシステムディスクをドライブ B：にセットして作業を始めます。

```
A>COPY MELEMM.SYS B:⏎ ……………EMMドライバをコピーする
        1 個のファイルをコピーしました．

A>DIR B:⏎ ……………ディスクの内容を確認する

 ドライブ B: のディスクのボリュームラベルはありません．
 ディレクトリは B:¥

COMMAND  COM     24931  88-10-16   16:10 ┐
VJEB     SYS     76913  89-01-31    2:00 │
VJEB     DRV      2763  88-10-10    2:00 │ 1.4章で述べたVJE-β用の各種ファイル
VJEB     DIC    488448  89-01-08    2:00 │
SETVJE   EXE     15394  88-11-01    2:00 │
CONFIG   SYS        84  89-09-07   23:33 ┘
MELEMM   SYS     16416  89-02-21   11:43 ……………MELWAREのオリジナルディスクから、
                                           EMSを設定するのに必要なファイルがコピーされた
        7 個のファイルがあります．
   525312 バイトが使用可能です．

A>
     ⋮
   途中省略(CONFIG.SYSファイルに、下記のデバイスドライバを
   エディタやCOPYコマンドを使って登録する)
     ⋮

B>TYPE CONFIG.SYS⏎ ……………EMSを設定するためのデバイスドライバを組み込んだ
                            CONFIG.SYSをタイプアウトする
FILES=10
BUFFERS=20
DEVICE = MELEMM.SYS /B120 ……………EMMドライバをEMSドライバとして機能させる。
                                  EMSには全バンクメモリ(1920Kバイト)を割り当てる
DEVICE = VJEB.DRV /HR /M1 /TO /L /J /K- /K, /K. /D4

B>
```

図 3.24　VJE を組み込んだ MS-DOS のシステムディスク上に EMS 拡張メモリを設定する

なお、CONFIG.SYS ファイルの書き換え（1行追加）は、エディタやワープロを使って行いますが、COPY コマンドを使って、最初から全体を作り直してもかまいません。

以上のように設定をした後、このシステムディスクで MS-DOS を再起動すれば、EMS によるバンクメモリが実現され、EMS 対応のアプリケーションプログラムの利用が可能になります。ただし、ここで使用している RAM ボード（HCE-2000）は、ハードウェア対応の EMS ボードではないため、MELWARE は、ソフトウェア的に EMS をエミュレートします。

■ EMS 拡張メモリを使ったアプリケーションプログラムの実行

さて、1920K バイトの EMS 拡張メモリが設定されたシステムディスクをドライブ A：にセットして起動すると、MS-DOS のオープニングメッセージとともに、EMM のデバイスドライバがシステムに組み込まれ、その際のメッセージが表示されます。

（リセットボタンを押す）

```
NEC PC-9800 Series Personal Computer

ﾏｲｸﾛｿﾌﾄ MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30
Copyright 1981,88 Microsoft Corp. / NEC Corporation

   》バンクメモリーマネジャー《      [Ver2.21] Jan.1989 MELCO Inc. Japan
   》    ＥＭＳドライバー    《      [Ver2.21] Jan.1989 MELCO Inc. Japan
   ※  ＥＭＳメモリーは  １１９  ページ 使用可能です。

日本語入力フロント・プロセッサ ＶＪＥ－β 第２．００版
Copyright (C) 1986-88 VACS Corp. / ASCII Corp.

Command ﾊﾞｰｼﾞｮﾝ 3.30
現在の日付は 1989-09-06 (水) です.
日付を入力してください:↵
現在の時刻は 22:34:30.00 です.
時刻を入力してください:↵

A>
```

MS-DOSのオープニングメッセージ

EMMドライバ／EMSドライバの組み込みメッセージ。1920KバイトのEMSメモリが設定されている*

VJE-βのFEPドライバ組み込みメッセージ

＊1ページ16Kバイトなので、本来は120ページが使用可能なはずだが、EMSドライバの管理領域として1ページ分が消費されるため、119ページとなる。

A> ……………システムが完全に立ち上がった。Lotus 1-2-3などのアプリケーションプログラムで1920KバイトのEMSメモリが使用できる

図 3.25　EMM が組み込まれたことを示す MS-DOS 起動時のメッセージ

このメッセージには、1904K バイトの EMS 拡張メモリが設定されたことなどが表示されています。現在、EMS 対応のアプリケーションプログラムの数は、まだ多くはありませんが、ここでは表計算ソフトの「1-2-3」を実行して、そこで作成する表の大きさの限界が、EMS 拡張メモリによって広がる様子を見てみましょう。ただし、表に書き込むデータの種類や値によっても、表が拡大される状態は大きく変わりますので、ここでの例はほんの一例であることをお断りしておきます。

〈EMS拡張メモリを使用しない場合〉

複写先を指定してください：C1656 ……小数点形式のデータを各列の最終行(8192行)までコピーしていく

エラー

	A	B	C	D	E	F	G
1650	1963.914	1963.914	1963.914				
1651	1963.914	1963.914	1963.914				
1652	1963.914	1963.914	1963.914				
1653	1963.914	1963.914	1963.914				
1654	1963.914	1963.914	1963.914				
1655	1963.914	1963.914	1963.914				
1656	1963.914	1963.914					
1657	1963.914	1963.914					
1658	1963.914	1963.914					
1659	1963.914	1963.914					
1666	1963.914	1963.914					
1667	1963.914	1963.914					
1668	1963.914	1963.914					
1669	1963.914	1963.914					

18,039個のセルまで使用可能であった。ここが最終セル

メモリがいっぱいです ……注目

メニュー 編集 再編集 絶対 ジャンプ カナ変換 半角変 ローマ字 半角 カナ

この先8192行まで、同一データがコピーされている

〈EMS拡張メモリを使用した場合〉

複写先を指定してください：I3005

エラー

	D	E	F	G	H	I	J
2991	1963.914	1963.914	1963.914	1963.914	1963.914	1963.914	
2992	1963.914	1963.914	1963.914	1963.914	1963.914	1963.914	
2993	1963.914	1963.914	1963.914	1963.914	1963.914	1963.914	
3000	1963.914	1963.914	1963.914	1963.914			
3001	1963.914	1963.914	1963.914	1963.914	1963.914	1963.914	
3002	1963.914	1963.914	1963.914	1963.914	1963.914	1963.914	
3003	1963.914	1963.914	1963.914	1963.914	1963.914	1963.914	
3004	1963.914	1963.914	1963.914	1963.914	1963.914	1963.914	
3005	1963.914	1963.914	1963.914	1963.914	1963.914		
3006	1963.914	1963.914	1963.914	1963.914	1963.914		
3007	1963.914	1963.914	1963.914	1963.914	1963.914		
3008	1963.914	1963.914	1963.914	1963.914	1963.914		
3009	1963.914	1963.914	1963.914	1963.914	1963.914		
3010	1963.914	1963.914	1963.914	1963.914	1963.914		

68,540個のセルまで使用できた！EMSを使用しない場合と比較して、約3.8倍のセルが使用できたことになる

メモリがいっぱいです ……注目

メニュー 編集 再編集 絶対 ジャンプ カナ変換 半角変 ローマ字 半角 カナ

この先8192行まで同一データがコピーされている

図 3.26 EMS 拡張メモリを使った 1-2-3 の実行

■ EMS拡張メモリ使用上の注意

EMSに使うためのRAMボードは、EMSによるバンク切り替えを、ハードウェアで行うように設計された**ハードウェアEMS方式**のRAMボードを選択することをお勧めします。「ハードウェアEMS方式」に対し、一般のバンク切り替えのRAMボード上(本節の実行例)、あるいはプロテクト増設メモリ上でEMSを実現した場合は、EMSの動作をソフトウェア的にエミュレートする(まねる)ことになるため、動作がかなり遅くなります。

また、EMSによるバンクメモリは、640Kバイトのユーザーエリア外の、たとえば、一般的にはアドレスC0000H(任意である)からの拡張ROM領域(PC-9800シリーズなどの場合)に設定されます。そこでもし、同じメモリ領域を使用しているほかの装置のボードなどが存在する場合、両者が競合することになり、どちらかの設定アドレスを変更しなければなりませんので注意が必要です。

さらに、プロテクト増設メモリでEMSを実現する場合(この場合は、ソフトウェア的にEMSをエミュレートする方式)、コンピュータのリセット時や電源ON時には、そのメモリ内容がすべてクリアされます。いわゆるウォームブートはできませんので、リセットには十分注意してください。

3.7 MELWAREの簡単設定プログラム

MELWAREのオリジナルディスクには、「MELCO.BAT」というバッチファイルが含まれています。このバッチファイルは、本章の3.4章以降で行った、RAMディスクやディスクキャッシュ、EMS拡張メモリなどを実現するための各種の設定を、メニュー方式で簡単に行うためのプログラムです。

このプログラムの主な機能は次の3つです。

①対象マシンのメモリの実装状況やディスクドライブの接続状態などのシステム環境を調べる
②要求された機能を実現するためのデバイスドライバのプログラムファイルを指定のディスク上にコピーする
③そのデバイスドライバを動作させるための指示と、各種の設定事項を、CONFIG.SYSファイルへ登録する

この簡単設定プログラム(バッチファイル)の実行は、表示されるメニューの指示にしたがって行っていけばよいだけですので、さっそく実行してみましょう。RAMボードのハードウェア的な設定は、バンク切り替えメモリとした場合の例を示します。

設定しようとする各種のアプリケーションプログラムなどのディスクは、B：あるいはC：となるようにディスクドライブを構成しておいてください。つまり、バッチファイル「MELCO.BAT」(および関係するファイルのすべて)をドライブA：、あるいはドライブB：に置き、そのドライブをカレントドライブとして実行します。

では、MELWAREのオリジナルディスク(もちろんバックアップコピーでよい)をドライブA：にセットして、簡単設定プログラムを実行し、RAMディスク、ディスクキャッシュ、EMSを同時に設定してみましょう。設定対象のディスクは、例によって1.4章で作成した、日本語入力システムのVJEを組み込んだMS-DOSシステムディスクとします。

なお、このバッチファイルの実行には、オリジナルディスクに含まれている各種のファイルが実行されたり参照されたりしますので、結局、MELWAREのオリジナルディスク、あるいはそのコピーをセットしておく必要があります。

「A＞MELCO↵」によりバッチファイルを実行して、次ページの図3.27のように、メニューにしたがった操作により、大まかな各種の設定を行うことができました(さきに行ったマニュアル設定の方が、よほどスッキリして、内容もわかりやすいと思うが)。この設定プログラムでは、細かい事項が設定できないものもありますので、必要であれば、エディタやワープロを使って、CONFIG.SYSファイル内の、さきに示した各パラメータを修正してください。

簡単設定プログラムによるメモリの設定

```
★　メモリ設定　★　《メルウェア Ver 2.21 Jan.1989 MELCO Inc. Japan 》
 Ⅰ．日本語処理のバンクメモリの使用量
   使用しない      VJE        松茸        ATOK
 Ⅱ．汎用RAMディスクの使用量（本体リセット時 データを初期化しない）
   使用しない                      640 Kバイト
 Ⅲ．キャッシュの使用量
   使用しない                      640 Kバイト
 Ⅳ．EMSの使用可能なメモリー量       640 Kバイト
   【 終了：ESC(ｴｽｹｰﾌﾟ)、 減少：←、 増加：→、 選択：↑↓ 】
```

FEPをメインメモリからバンクメモリに追い出す場合に設定

RAMディスク、ディスクキャッシュ、EMSへのメモリの割り当てを決める

簡単設定プログラムを終了する

```
B>DIR
ドライブ B: のディスクのボリュームラベルはありません．
ディレクトリは B:¥

COMMAND  COM     24931  88-10-16  16:10
VJEB     SYS     76913  89-01-31   2:00
VJEB     DRV      2763  88-10-10   2:00
VJEB     DIC    488448  89-01-08   2:00
SETVJE   EXE     15394  88-11-01   2:00
CONFIG   ORG        84  89-09-07  23:37
CONFIG   SYS       176  89-09-07  23:53
MELEMM   SYS     16416  89-02-21  11:43
MELDISK  SYS      5888  89-01-05  22:00
MELCACHE SYS      9253  89-01-18  15:02
    10 個のファイルがあります．
  507904 バイトが使用可能です．

B>TYPE CONFIG.SYS
FILES=10
BUFFERS=20
DEVICE = MELEMM.SYS   /B40 /S5
DEVICE = MELDISK.SYS   640  /S
DEVICE = MELCACHE.SYS  640
DEVICE = VJEB.DRV /HR /M1 /TO /L /J /K- /K, /K. /D4

B>
```

B>DIR ……簡単設定プログラムで作成されたディスクの内容を確認する

1.4章で述べたVJE-β用の各種ファイル

すでにあったCONFIG.SYSファイルはリネームされて残されている

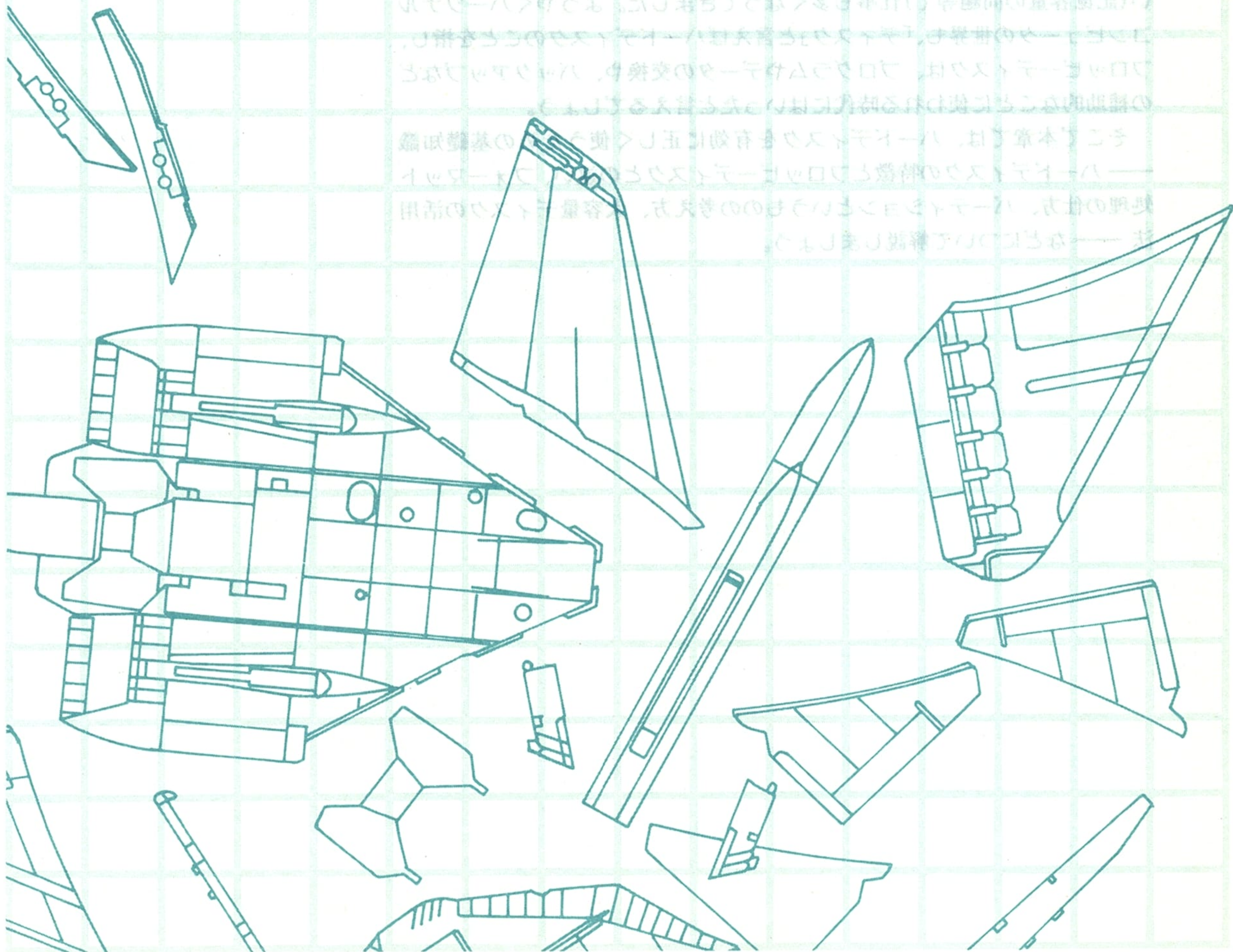
必要なデバイスドライバを追加登録したCONFIG.SYSが作成されている

必要なデバイスドライバはすべてコピーされている

B>TYPE CONFIG.SYS ……簡単設定プログラムで作成されたCONFIG.SYSの内容をタイプアウトする

これらのデバイスドライバが追加登録されている。設定を変える場合は、これらのパラメータをエディタなどで直接変更してもよい

図3.27　簡単設定プログラムを使って、RAMディスク、ディスクキャッシュ、EMSの3つが同時に設定されたディスクを作成する

4章
ハードディスクの使い方

1988～89年を境として、パーソナルコンピュータのディスク装置も、本格的なハードディスク時代を迎えました。それ以前、ハードディスクはかなり高価な装置でしたが、そのわりに記憶容量は、1ドライブあたり、せいぜい10M～20Mバイト程度のものでした。それが今日、価格はフロッピーディスクドライブに接近するほどまでに低下し、記憶容量も40Mバイト程度は普通となり、さらには100Mバイト以上のものまでが使われるようになりました。この背景には、パーソナルコンピュータが実務に広く使われるようになるにともない、取り扱うデータ量の増加や、仕事の能率向上の面から、必然的に高速・大容量のディスク装置が求められるようになったことが考えられます。

このようなことから、ハードディスクは広く一般に、急速に使われるようになりました。もはや本格的な実務には、ハードディスクを使うことが常識となり、現実にも、ハードディスクでなければ、物理的に処理できない(記憶容量の問題等で)仕事も多くなってきました。ようやくパーソナルコンピュータの世界も、「ディスク」と言えばハードディスクのことを指し、フロッピーディスクは、プログラムやデータの交換や、バックアップなどの補助的なことに使われる時代にはいったと言えるでしょう。

そこで本章では、ハードディスクを有効に正しく使うための基礎知識――ハードディスクの特徴とフロッピーディスクとの違い、フォーマット処理の仕方、パーティションというものの考え方、大容量ディスクの活用法――などについて解説しましょう。

4.1 ハードディスクの特徴

ハードディスクの特徴としては、まず「高速・大容量」があげられますが、「信頼性、耐久性」の面でも非常に優れています。現在のところ、一般的なコンピュータの主外部記憶装置として、このハードディスクにとって代わるような、種々の条件を満足する記憶装置は見あたりません。ハードディスクは、現在のすべてのコンピュータシステムにおける外部記憶装置の中心であり、〝標準装置〟(キーボードやディスプレイのような)と言ってもよいほど、なくてはならない重要な装置です。このハードディスクの優れた性能や機能は、フロッピーディスクとは異なる機械的な構造により実現されていますが、その概要を簡単に見ていきましょう。

高速・大容量

まず、ディスクアクセス(読み書きの動作)が高速であることと、記憶容量が大きいことについて考えてみましょう。一般的なフロッピーディスクの場合、記憶容量は約1.2Mバイト(5インチや3.5インチ2HDの場合)、データ転送速度は約300Kbit/sec(1秒間に300Kビットの情報を読み書きする)程度です。フロッピーディスク(メディア)とディスクドライブとは、互いに分離していますので、メディア(磁気記録媒体)の交換が自由に行えます。メディア自体は柔軟なフィルムシートであり、そのため、フロッピーディスクの正式名は「フレキシブルディスク」と呼ばれています。また、IBM社の商品名として、「ディスケット」とも呼ばれます。

一方ハードディスクの場合、記憶容量は普通のもので40Mバイト程度、大容量のものでは100Mバイト以上のものまで使われ、データ転送速度は一般的なものでも5Mbit/sec程度と非常に高速です(パーソナルコンピュータ用の場合)。ハードディスクは、別名「固定ディスク」とも呼ばれているように、メディアは硬い円板であり、それがディスクドライブと一体構造になっているため、特殊なものを除いてメディアの交換はできません(大型コンピュータやミニコンピュータに使われているハードディスクでは、メディアの交換が可能な機種もある。パーソナルコンピュータ用のものでは、メディアとヘッド周辺の機構とが一体になったものを交換できる機種がある)。

硬い円盤であることと、それが装置と一体に取り付けられて回転するために、フロッピーディスクに比べ、各部の機械的な精度や、磁性面の精度を上げることができ、かつ高速回転が可能です。つまり、記録の高密度化と、リード／ライトの高速化が実現できるわけです。また記憶容量は、複数の円盤を積み重ねることによって増やすことができますので、汎用コンピュータに使われるものでは、1台で1000Mバイト程度の大容量のものまで実用化されています。

ただし、MS-DOSマシンで管理できる容量は、ディスクインターフェイスによる制限や、各社が作成するMS-DOSシステム内の「IO.SYS」やFORMATコマンドなどのプログラムの都合上、現在のところ、1ドライブあたり最大128Mバイト程度(MS-DOSバージョン3.*x*の場合)になっています。

高耐久性／高信頼性

次に、耐久性と信頼性について考えてみましょう。この両者は密接に関係していて、分けて考えることはできません。フロッピーディスクと比較して、耐久性や信頼性に関係すると思われるメカニズムのもっとも大きな違いは2つあります。まず1つは、回転する磁性面と、データを読み／書きするヘッドとが物理的に接触しないこと、もう1つは、それらが外の空気に直接さらされていないことの2点です。

フロッピーディスクの場合は、データの読み／書きを行うとき、ヘッドが磁気円盤シートに物理的に接触します(カタカタという動作音はそのときの音。ビービーという音はヘッドが移動する音)。ですから、わずかではあっても、そのたびに磁性面やヘッドが摩耗することは避けられません。また物理的に磁性面をヘッドが「こする」わけですから、ヘッドにゴミや汚れが付着する危険性も十分に考えられることです。

一方、ハードディスクの磁気円盤は、普通、毎分3600回転(フロッピーディスクの10倍)の非常な高速で回転しているため、円盤にごく近い表面には、非常に高速の空気流が発生しています。ディスクが動作状態にあるとき、ヘッドは、バネの力などで円盤に軽く押しつけられていますが、空気の流体力学的な力によって、円盤表面から1ミクロン以下のオーダー(約0.1〜0.2ミクロン程度。1ミクロンは1/1000㎜)で浮き上がります。つまり、ヘッドは円盤の磁性面に接触せず、データの読み／書きを行うわけで、磁性面やヘッドが摩耗することはありません。

ハードディスクの取り扱いで、アクセス中(読み／書きの動作中)に電源を絶対にOFFにしてはいけないわけは、ディスクの回転数の低下とともに、ヘッドを浮上させる力が失なわれ、ついにはヘッドが回転する円盤に接触して、ヘッドも磁性面も致命的なダメージを受けることになるからです。また、装置に強い衝撃を与えることによっても、ヘッドが円盤に接触して、同様のダメージを受けます。このようなダメージは**ヘッドクラッシュ**と呼ばれ、非常に恐ろしいものですので、電源のOFFや衝撃には細心の注意が必要です。

電源をOFFにする際には、まずMS-DOSのコマンドレベルに下(お)りて、STOPキーを押します。そうすることにより、ヘッドが円盤の表面から持ち上げられ、所定の位置(ホームポジション)にもどります。そのあとで(STOPキーにより、ディスクのアクセスランプが点灯しますので、それが消えたのを確認してから)電源をOFFにするようにしてください。なお、各機種のMS-DOSシステムディスクや、アプリケーションプログラムによっては、一定時間以上ハードディスクが使われない場合、自動的にこの処理を行うものもあります。

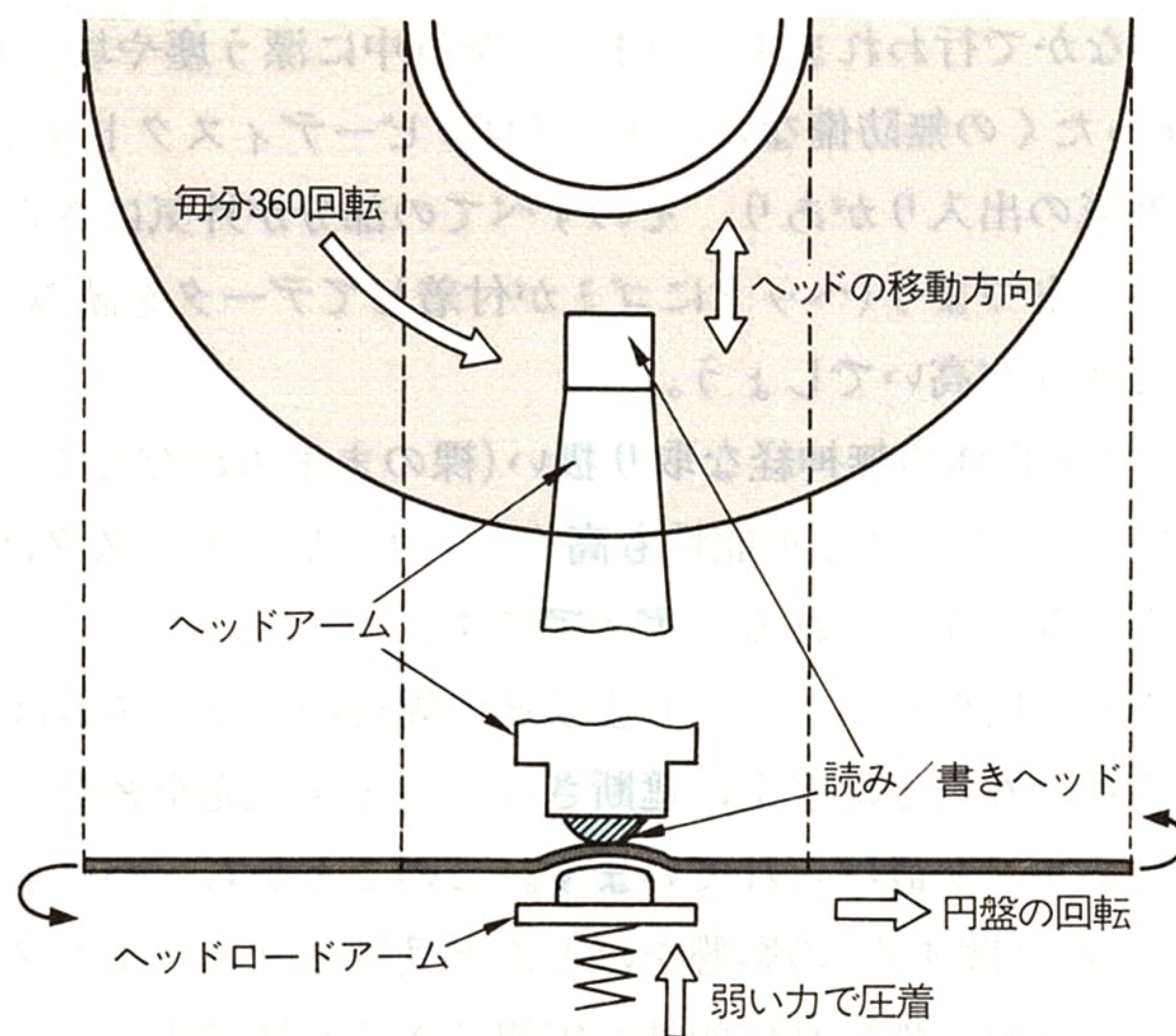
このように、ハードディスクとフロッピーディスクとでは、データをリード／ライトする際のヘッドとディスクとの関係に大きな違いがありますが、同時に、ヘッドとディスクの周囲の環境も大きく異なっています。

読み書きを行うときは、ヘッドロードアームがディスクに圧着され、ディスクとヘッドが接触する

図 4.1　ヘッドとディスクとの関係　フロッピーディスクの場合

円盤の高速回転により、円盤表面に空気流が発生し、その流体力学によりヘッドが浮上する。そのすき間はおよそ 0.1〜0.2ミクロン。

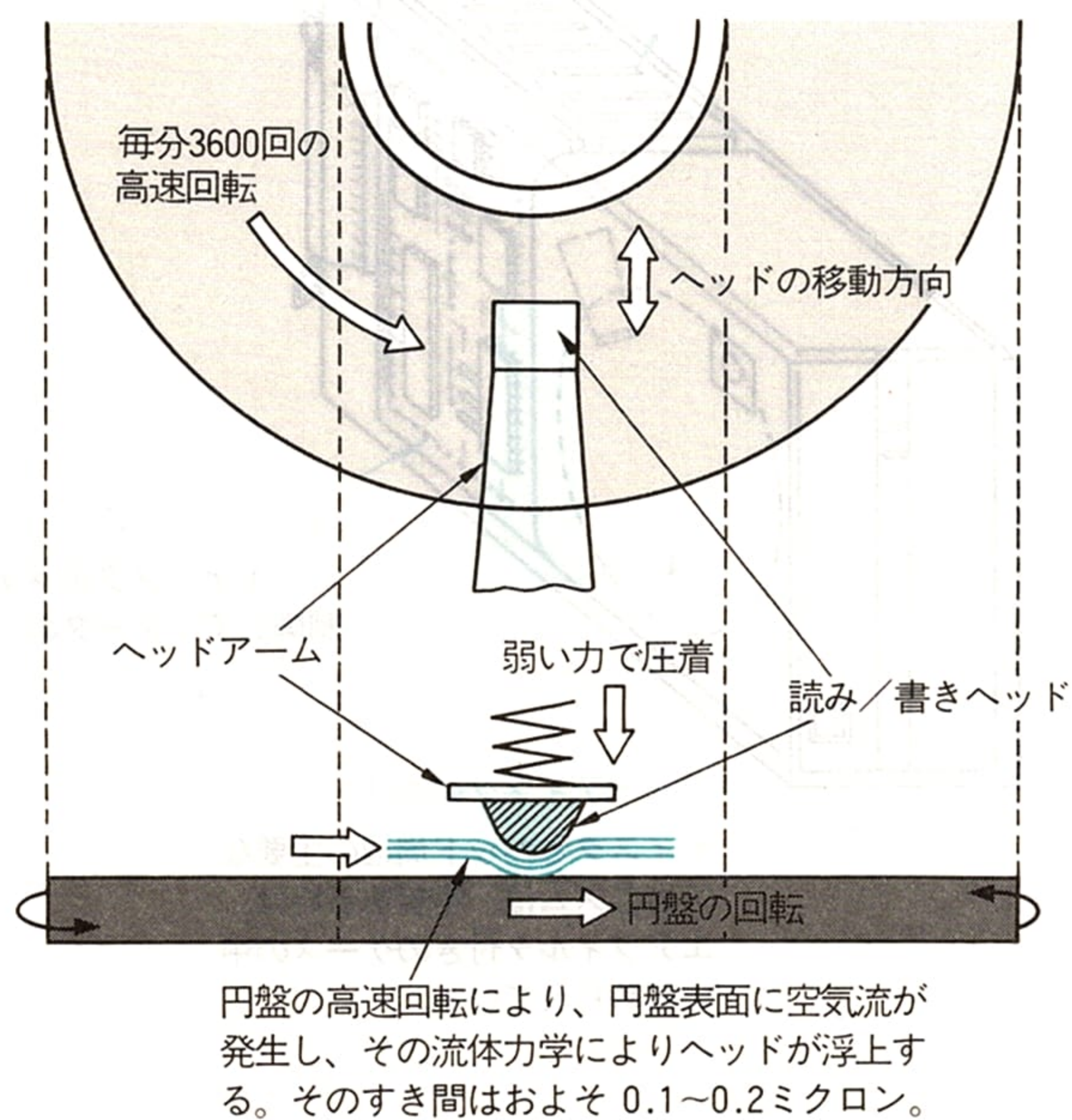



図 4.2　ヘッドとディスクとの関係　ハードディスクの場合

まずフロッピーディスクの場合は、メディアもヘッドもメカニズムも、外気に対しては「むき出し」の状態であり、動作もそのなかで行われます。つまり、空気中に漂う塵(ちり)や埃(ほこり)、タバコの煙(実体は液状の粒子)などの汚染にはまったくの無防備なのです。フロッピーディスクドライブの内部は、空冷ファンや、熱対流などによる外気の出入りがあり、そのすべての部分が外気にさらされます。その結果、使用環境が悪い場合は、ヘッドづまり(ヘッドにゴミが付着してデータを読み／書きできない状態)などのトラブルが発生する可能性が高いでしょう。

さらに、フロッピーディスク自体の無神経な取り扱い(裸のまま汚れたところへ置いたり、無理な力を加えたりなど)によりダメージを受ける可能性も高く、フロッピーディスクは、使用環境や取り扱いにより、耐久性や信頼性が大きく左右されるメディアです。

一方、ハードディスクの磁気円盤やヘッド、およびその周辺のメカニズムは、それらの全体を収容するケースによって密封され、外気とは完全に遮断されています。完全密封でない構造の場合は、その通気部には厳重なエアフィルタが設けられています。このことから、ハードディスクの主要メカニズムは、使用環境(空気の汚れに関する)の影響をほとんど受けません。またメディア自体を取り出して交換することもないため、高耐久性、高信頼性が実現できるわけです。

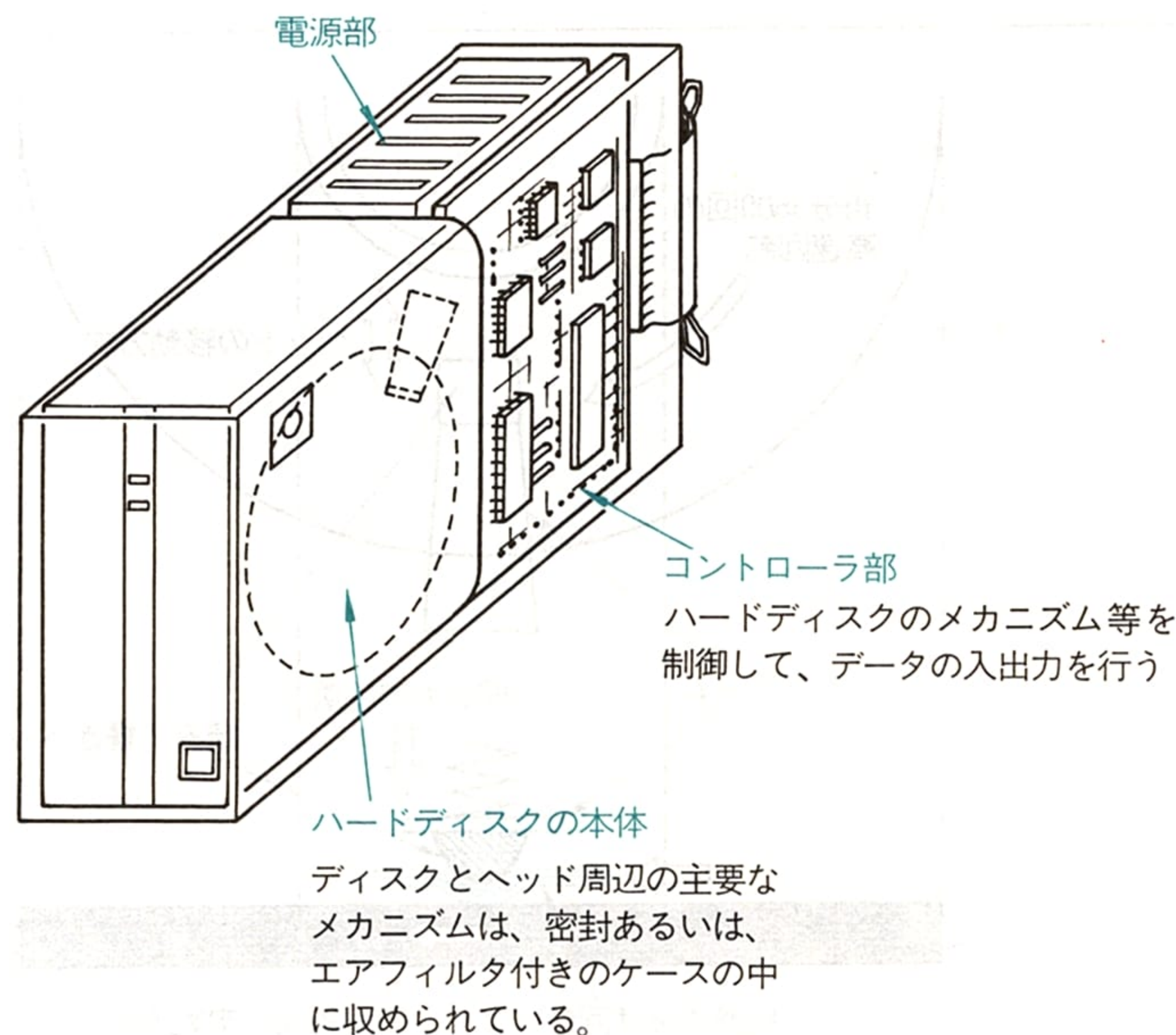
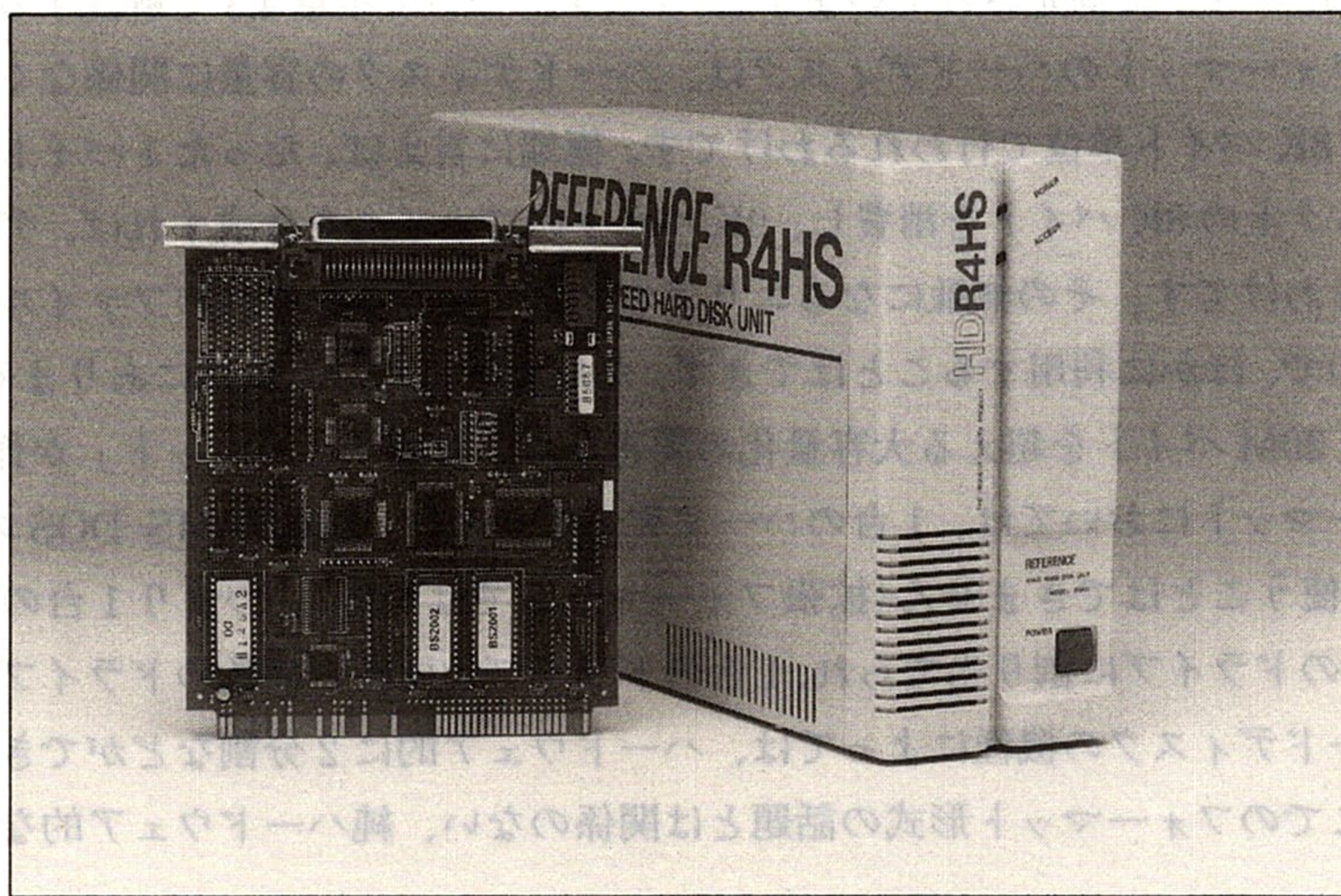



図 4.3　外気から守られているハードディスクの主要メカニズム

4.2 ハードディスクの初期設定

ハードディスクは、基本的には記憶容量が大きい高速フロッピーディスク(もちろん交換はできない)と考えてよいでしょう。ハードディスクであるからといって、ファイルの読み／書きや、階層ディレクトリによるファイル管理などの基本に違いがあるわけではありません。ただし、MS-DOSがファイルを管理する最小単位である「クラスタサイズ」は、1Mタイプのフロッピーディスクでは1Kバイトですが、ハードディスクでは、その初期設定により異なり、容量が40Mバイトのディスクでは、最大で16Kバイトになる場合もあります。この点は、従来のフロッピーディスクとは違った認識が必要です(くわしくは後述)。

このようにハードディスクは、使用上は〝高速大容量のフロッピーディスク〟にすぎませんが、フロッピーディスクの「フォーマット処理」(初期化)に相当するハードディスクの初期設定は、MS-DOSのバージョンやその提供メーカーの違い、フォーマット形式やハードディスクのインターフェイス規格の違いによって大幅に異なります。

ハードディスクの初期設定は、フロッピーディスクのフォーマット処理と同じフォーマットプログラム「FORMAT.EXE(.COM)」で行いますが、本節では、ハードディスクを実際に使えるようにするための〝難関〟である、初期設定について解説しましょう。なお、ここで示す実行例は、PC-9800シリーズにおける標準的なハードディスクを使った場合のものです(他社メーカーのハードディスクも、機種によってはまったく同様に使えるものがあります)。

40Mバイトのハードディスクとインターフェイスボード(ウィンテク社)

■ MS-DOS バージョン 2.x でハードディスクを使う場合

まずハードウェア的には、ハードディスクのインターフェイスは「SASI（サシー）」規格のもののみが使用可能であり、「SCSI（スカジー）」規格のインターフェイスは使用できません（SASI／SCSI：後述）。また、管理可能なディスク容量と台数は、このあとに述べるように「最大20Mバイト×2台」ですので、それ以上の容量のディスクを接続しても、1台あたりは20Mバイトとしてしか使えません。

ソフトウェア的には、ハードディスクを初期設定する際のフォーマットの形式に制約があります。ハードディスクのフォーマットには、**標準フォーマット**と**拡張フォーマット**と呼ばれる2つの形式がありますが、MS-DOSのバージョン2.*x*では、「標準フォーマット」のみが可能です。「拡張フォーマット」は、MS-DOSのバージョン3.*x*以降に対応したもので、逆に言えば、大容量のハードディスクに対応できる「拡張フォーマット」形式を採用するために、バージョンを3.*x*にアップした、ともいえるのです。

MS-DOS のバージョン 2.*x* では、「標準フォーマット」のみが使用可能である

標準フォーマット

ハードディスクの初期設定における「標準フォーマット」と呼ばれる形式は、MS-DOSバージョン2.*x*時代の形式であり、管理可能なハードディスクの容量に、1台あたり最大20Mバイトという制限があります。また、接続可能な台数は、ディスクインターフェイスの制限から2台までとなります。

標準フォーマットにおける1クラスタに対応するFAT（ファイルアロケーションテーブル。各ファイルのデータが格納されている場所を管理するためのテーブル）は12ビットであり、また、1クラスタのサイズは常に8Kバイトに固定されています（クラスタについて本章4.3で解説する）。

つまり、標準フォーマットのハードディスクは、ハードディスクの容量に関係なく、ファイルの書き込みや削除は、8Kバイト単位で行われるわけです。極端に言えば、たった1バイトのファイルを作成しても、ディスク上の8Kバイトを消費し、9Kバイトのファイルを作成すれば、ディスク上の16Kバイトを消費するわけです。その無駄になるエリアも、あくまでそれぞれのファイルのために確保されたエリアですので、ほかに利用することはできず、ただ放置しておく以外にありません（無駄になるエリアの軽減と、20Mバイトを超える大容量化の要求から「拡張フォーマット」が作られた）。

また標準フォーマットにおいては、1台のハードディスク内を、複数のMS-DOSの領域（**パーティション**）に分けて使うことはできません（拡張フォーマットでは可能）。つまり1台のハードディスクは、あくまで1つのドライブに割り当てられ、1台のドライブが同時に複数のドライブ名を持つことはできません。ハードディスクの機種によっては、ハードウェア的に2分割などができるものもありますが、それはここでのフォーマット形式の話題とは関係のない、純ハードウェア的な問題です。

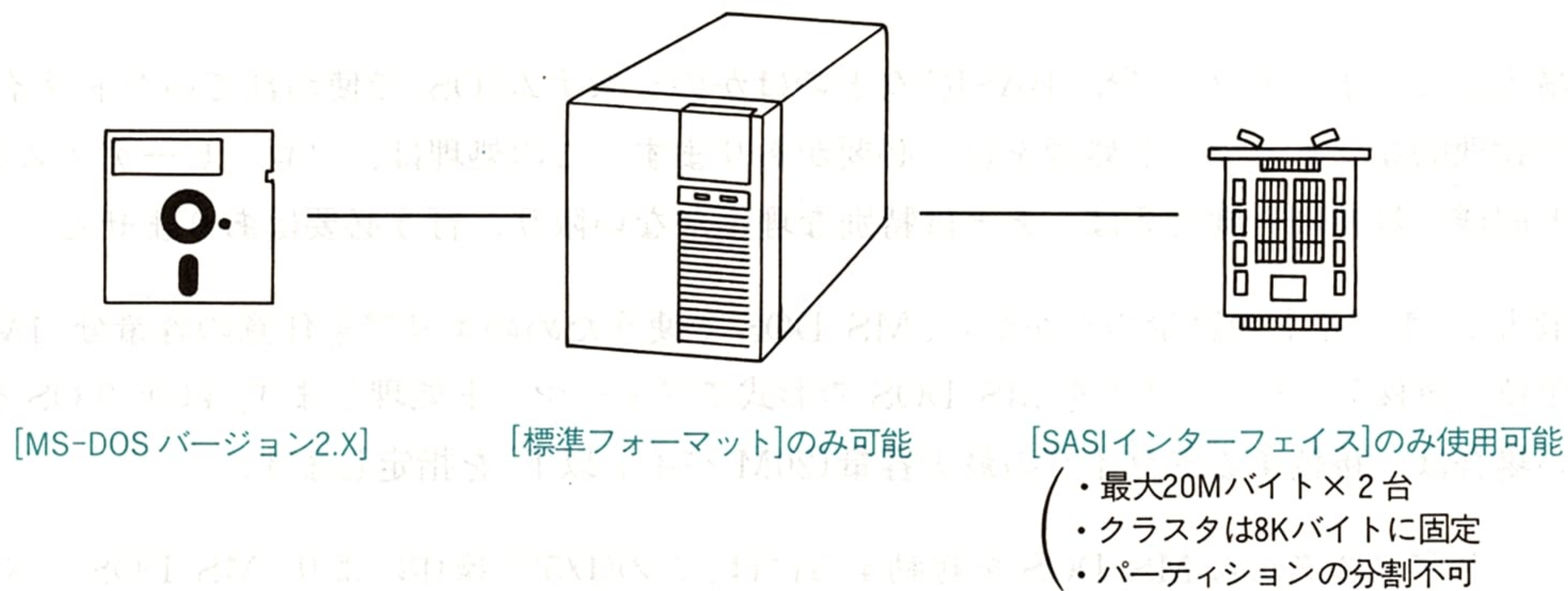



図 4.4　MS-DOS バージョン 2.x における接続可能なハードディスク

標準フォーマットによる初期設定

では実際に、標準フォーマットによるハードディスクの初期設定を行ってみましょう。標準フォーマットの場合、初期設定は次のような手順で行います。

FORMAT コマンドを起動

↓

① ハードディスクの「装置番号」(ドライブ番号 1、または 2)を指定する

↓

②「初期化」(ドライブ全体の物理的なフォーマット処理)を行う

↓

③「領域確保」(MS-DOS が使用するエリアの論理的なフォーマット処理)を行う

↓

④「システム」(MS-DOS システム)をコピーする

↓

⑤「IPL」(イニシャルプログラムローダ)を登録する

↓

⑥「ボリュームラベル」を登録する

（④～⑥：必要であれば設定する）

↓

終わり

①：接続可能な2台のドライブの、どちらを初期設定するかの指定です。主ドライブが「1」、増設ドライブが「2」となります。

②：購入したままのドライブや、BASICなどのほかのシステム(OS)で使われていたドライブは、この物理的なフォーマット処理を行う必要があります。この処理は、フロッピーディスクの場合と同様、最初に一度行えば、あとは特別な理由がない限り、行う必要はありません。

③：最大20Mバイトの容量のなかから、MS-DOSで使うためのエリアを任意の容量分、1Mバイト単位で確保し、そのエリアをMS-DOSの形式でフォーマット処理します。ほかのOSを使わない場合は、接続するドライブの最大容量(20Mバイト以下)を指定します。

④⑤：ハードディスクからMS-DOSを起動するには、この④⑤の操作により、MS-DOSシステムと、その起動のためのプログラム(IPL)を登録しておく必要があります。ただしMS-DOSの起動は、増設ドライブ側からは行うことができませんので、システムやIPLを「2」側に登録することには意味がありません。なお、ハードディスクからMS-DOSを起動しない場合は、この部分の操作は必要ありません。

⑥：ボリュームラベルが必要なら、ここで付けることができます。

では、20Mバイトのハードディスクを、MS-DOSが起動できる20Mバイト(全部の領域を使った場合)のドライブにするための、まったく最初からの初期設定の実行例を示しましょう。

ドライブA：にMS-DOSのシステムディスクをセットして、接続したハードディスクの電源を必ずONにした後、MS-DOSを立ち上げてください。MS-DOSがすでに立ち上がっているあとから、ハードディスクの電源をONにしても、MS-DOSは、ハードディスクが接続されていることを認識しませんので注意が必要です。

ハードディスクの初期設定は、フロッピーディスクと同じFORMATコマンドで行いますので、そのプログラムファイル「FORMAT.COM」(MS-DOSのバージョン3.*x*ではFORMAT.EXE)が必要です。確認しておいてください。なお、この節に限り、MS-DOSのバージョン2.*x*を使った実行例を示しますが、バージョン3.*x*の場合も、「標準フォーマット」を選択することにより、ほとんど同様に実行することができます(作業の内容は同じであるが、画面の表示は異なる)。

では始めましょう。ハードディスクの初期設定には、次のようにスイッチ「/H」を付けてFORMATコマンドを実行します。フロッピーディスクのフォーマット処理の場合には必要な、目的のドライブ名の指定は不用です。

```
A>FORMAT /H ⏎
```

スイッチ「/H」によって、フロッピーディスクのフォーマット処理とは違った、ハードディスク専用の初期設定モードにはいり、次のような画面が表示されます。実際の操作については、画面のなかで示しましょう。

```
A>FORMAT /H ⏎ ……………MS-DOSバージョン2.1のFORMATコマンドを起動して、
                     ハードディスクの初期設定を行う

Format  Version 2.50

固定ディスクの装置番号を入力してください [1,2] = 1⏎ ………1台目のハードディスクか、2台目の
                                                     ハードディスクかを選択する
モードを選択してください
1:マップ 2:領域確保 3:領域解放 4:IPL 5:装置初期化 6:装置変更 7:終了 = 5⏎
                                        ↑ハードディスクの各種設定   ↑ハードディスクの物理
                                         のためのメニュー            的な初期化（フォーマ
                                                                    ット処理）を選択する
        現在の装置使用状況        装置番号 1

        MS−DOS                 領域無し ……………購入直後、まだ一度もフォーマット処理を
        他のOS                  領域無し           していない場合、このように表示される
        空き領域                領域無し

装置全体を初期化します、いいですか <Y/N> ? Y⏎

装置を初期化中です        } 20Mバイトの場合、約4分ほど時間がかかる
  残り  0 メガバイト

モードを選択してください
1:マップ 2:領域確保 3:領域解放 4:IPL 5:装置初期化 6:装置変更 7:終了 = 1⏎

        現在の装置使用状況        装置番号 1

        MS−DOS                 領域無し
        他のOS                  領域無し
        空き領域                20 MB ……………20Mバイトの領域が初期化された

モードを選択してください
1:マップ 2:領域確保 3:領域解放 4:IPL 5:装置初期化 6:装置変更 7:終了 = 2⏎
                                                                  ↑
        現在の装置使用状況        装置番号 1                     MS-DOSで使用する
                                                                 領域を確保する
        MS−DOS                 領域無し
        他のOS                  領域無し
        空き領域                20 MB

領域の大きさを入力してください [1-20 MB] = 20⏎ ……………MS-DOSで使用する容量を指定する

領域確保中です            } 20Mバイトの場合、約6分ほど時間がかかる
  残り  0 メガバイト
```

```
システムを転送しますか <Y/N> ? Y⏎
システムを転送しました

IPLを登録しますか <Y/N> ? Y⏎
IPLを登録しました

ディスクのボリュームラベルを入力してください
漢字 - 5文字、英数字 - 11文字、無名称 - リターンキー  MS-DOS V2_1⏎

モードを選択してください
1:マップ 2:領域確保 3:領域解放 4:IPL 5:装置初期化 6:装置変更 7:終了 = 1⏎

          現在の装置使用状況              装置番号 1

          MS－DOS                         20 MB
          他のOS                          領域無し
          空き領域                        領域無し

モードを選択してください
1:マップ 2:領域確保 3:領域解放 4:IPL 5:装置初期化 6:装置変更 7:終了 = 7⏎
A>
```


図4.5　標準フォーマットにおける20Mバイトのシステムディスクの設定例

以上で、目的とする、20MバイトのMS-DOSが起動できるハードディスクの初期設定の作業が終わりました。ただしハードディスクは、このままの状態では動作しません。必ずMS-DOSを再起動する必要があります。

では、ドライブA：のシステムディスクを取り出した後、リセットボタンを押して、設定したハードディスクからMS-DOSを立ち上げてみましょう(PC-9801では、フロッピーディスク／ハードディスクのどちらからもMS-DOSの立ち上げを可能にするには、SWITCHコマンドの[BOOT]の設定を「STD」(標準)にしておく)。


```
NEC PC-9800 Series Personal Computer

ﾏｲｸﾛｿﾌﾄ MS-DOS ﾊﾞｰｼﾞｮﾝ 2.11
Copyright 1981,82,83 Microsoft Corp. / NEC Corporation

Command ﾊﾞｰｼﾞｮﾝ 2.11
現在の日付は 1989-09-16 (土) です.
日付を入力してください:⏎
現在の時刻は 16:43:43.00 です.
時刻を入力してください:⏎
```

```
A>DIR ⏎ ……………ハードディスクのディレクトリを確認する

 ドライブ A: のボリュームラベルは MS-DOS V2_1 ……………FORMATプログラムで設定したボリューム
 ディレクトリは A:¥                                 ラベルが設定されている

COMMAND  COM    17190  86-06-10   0:00 ……………システムを転送したので、COMMAND.COM
        1 個のファイルがあります                    がコピーされている
 20127744 バイトが使用可能です ……………当然ながら約20Mバイトの容量がある

A>
```

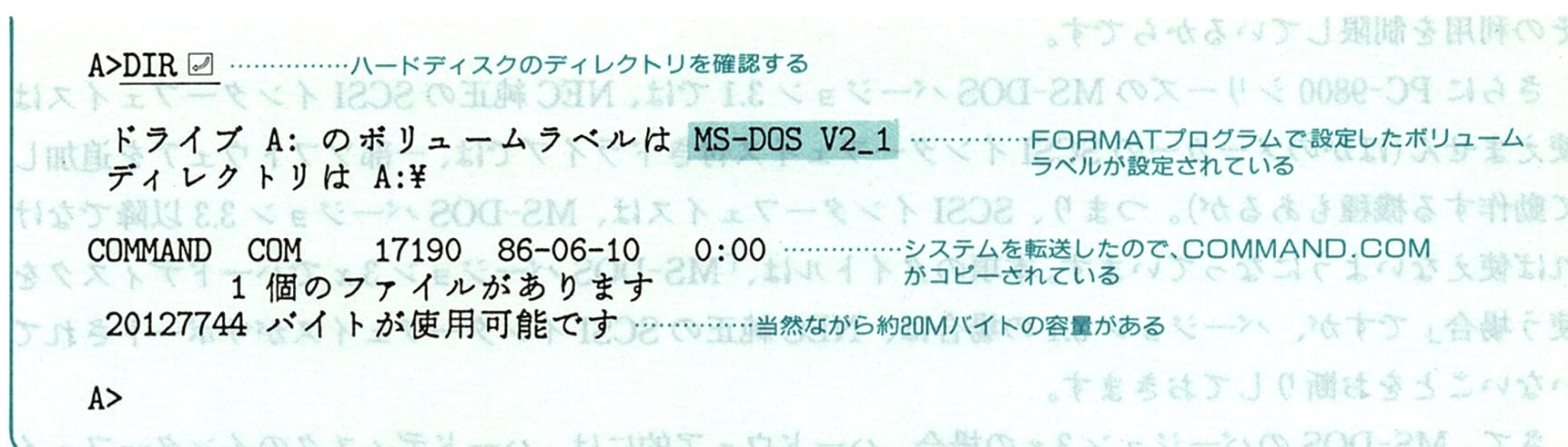



図 4.6　初期設定が完了したハードディスクから MS-DOS を立ち上げる

このように、ハードディスクから MS-DOS が起動しました。COMMAND.COM が 1 つだけコピーされていますが、これはさきの④にあたる「システムの転送」において、MS-DOS システムとともにコピーされたものです。

これで標準フォーマットによる 20M バイトのハードディスクが、完全に使える状態になりました。このあとは各ユーザーが、20M バイトという大容量のディスクを、いかにうまく利用するかという問題になります。

■ MS-DOS バージョン 3.x でハードディスクを使う場合

MS-DOS のバージョン 2.*x* から 3.*x* へのバージョンアップで追加された機能のうち、ハードディスクに関係するもっとも重要なものは、**16 ビット FAT** がサポートされたことです。バージョン 2.*x* では 12 ビットであった FAT が 16 ビットに拡張され、より大きな容量のハードディスクに対応できると同時に、クラスタのサイズを小さくすることが可能になったため、ディスクのメモリ容量の使用効率も向上しました(この件についてくわしくは、本章の 4.4 で解説)。

ただし、MS-DOS そのものは、バージョン 3.*x* 以降であれば、16 ビット FAT がサポートされているものの、それを利用する／しないは、各メーカーによって事情が異なります。たとえば PC-9800 シリーズの MS-DOS では、SCSI(スカジー)* インターフェイスで接続したハードディスクに対してのみ、16 ビット FAT が有効になります。つまり、SASI(サシー)* インターフェイスでは、せっかくの 16 ビット FAT は、まったく活用されないわけです。これは、各メーカーが、従来のディスク装置との互換性の問題などから、安全を考慮して、FORMAT コマンドや、MS-DOS システムの IO.SYS のドライバの作り方などで、

SASI：SASI (Shugart Associates System Interface) とは、米国のディスク装置メーカーの老舗であるシュガート社が開発した、コンピュータとハードディスク間のインターフェイス規格のことであり、PC-9800 シリーズでは、PC-9801-27 インターフェイスボードが、この規格に準拠したボードである。

SCSI：SCSI (Small Computer System Interface) とは、SASI をもとに ANSI (米国規格協会) が標準化した、パーソナルコンピュータの各種周辺装置の標準インターフェイス規格のことであり、PC-9800 シリーズでは、PC-9801-55 インターフェイスボードが、この規格に準拠したハードディスクのインターフェイスボードである。

その利用を制限しているからです。

さらにPC-9800シリーズのMS-DOSバージョン3.1では、NEC純正のSCSIインターフェイスは使えません(ほかのメーカーのSCSIインターフェイス付きドライブでは、一部ソフトウェアを追加して動作する機種もあるが)。つまり、SCSIインターフェイスは、MS-DOSバージョン3.3以降でなければ使えないようになっています。本項のタイトルは、「MS-DOSバージョン3.*x*でハードディスクを使う場合」ですが、バージョン3.1の場合は、NEC純正のSCSIインターフェイスがサポートされていないことをお断りしておきます。

さて、MS-DOSのバージョン3.*x*の場合、ハードウェア的には、ハードディスクのインターフェイスは「SASI」規格、および「SCSI」規格のインターフェイスが使用可能です。また、管理可能なディスク容量は、基本的には制限がないのですが、実際にはディスクのインターフェイスやFORMATコマンドのプログラム、MS-DOSシステムのIO.SYSのドライバの作り方などにより制約を受けます。

ソフトウェア的には、ハードディスクを初期設定する際のフォーマットの2つの形式——「標準フォーマット」と「拡張フォーマット」を任意に選択することができます。その「標準フォーマット」については、前節で解説しましたが、これは、MS-DOSのバージョン3.*x*においても同様です(作業の内容は同じであるが操作画面は異なる)。

たとえば、20Mバイトのハードディスクを、SASIインターフェイスで、20Mバイトのドライブとして使う場合は、標準フォーマットであっても、拡張フォーマットであっても、データの読み／書きに関する優劣はほとんどありません(クラスタサイズはどちらも8Kバイトで同じ)。しかし同じ条件でSCSIインターフェイスを使う場合は、拡張フォーマットを選択する方が、ディスクの使用効率の面で有利になります(クラスタサイズは2Kバイトになる)。この件については、本章の4.4で解説しますが、一般的には、拡張フォーマットで初期設定を行っておく方が、将来のためにも有利でしょう。

拡張フォーマット

ハードディスクの初期設定における「拡張フォーマット」と呼ばれる形式は、MS-DOSバージョン3.*x*でサポートされた形式であり、管理可能なハードディスクの容量に制限がありません。ただし実際にはさまざまな制約を受け、SASIインターフェイスの場合、最大40Mバイト×2台、SCSIインターフェイスの場合、最大128Mバイト×4台となります。

これらの制約は、さきに述べたように、MS-DOSのファイル管理における本質的な制約ではなく、現在のインターフェイスやFORMATコマンド、あるいはMS-DOSシステムの「IO.SYS」内のドライバ(各メーカーが独自に作成する)などの制約によるものであり、それらが変更されれば、この制約も変わる可能性があります。

拡張フォーマットにおける1クラスタに対応するFATは、従来の12ビットを拡張した16ビットを使うことが可能です。PC-9800シリーズでは、SCSIインターフェイスの場合に限り、11Mバイト以上の領域を確保すると、自動的に16ビットFATが使われるようにFORMATコマンドのプログラ

ムが作られています。これにより、大容量のハードディスクに対応すると同時に、ディスクのメモリの有効利用が図られます*。

拡張フォーマットのもう1つの大きな特徴は、1台のドライブを、最大4つのMS-DOSの領域(**パーティション**)——4つの〝ドライブ〟と考えてよい——に分けて使うことができることです(分けるだけであれば8つ)。1台のドライブを、各パーティションに分割することにより、MS-DOSの複数の〝別ドライブ〟として使ったり、ほかのOSと共用したりすることができるのです(くわしくは後述)。

ではここで、PC-9800シリーズのMS-DOSバージョン3.3における、接続可能なハードディスクの諸元を、フォーマット形式やインターフェイス形式の違いと関連させて示しておきましょう。

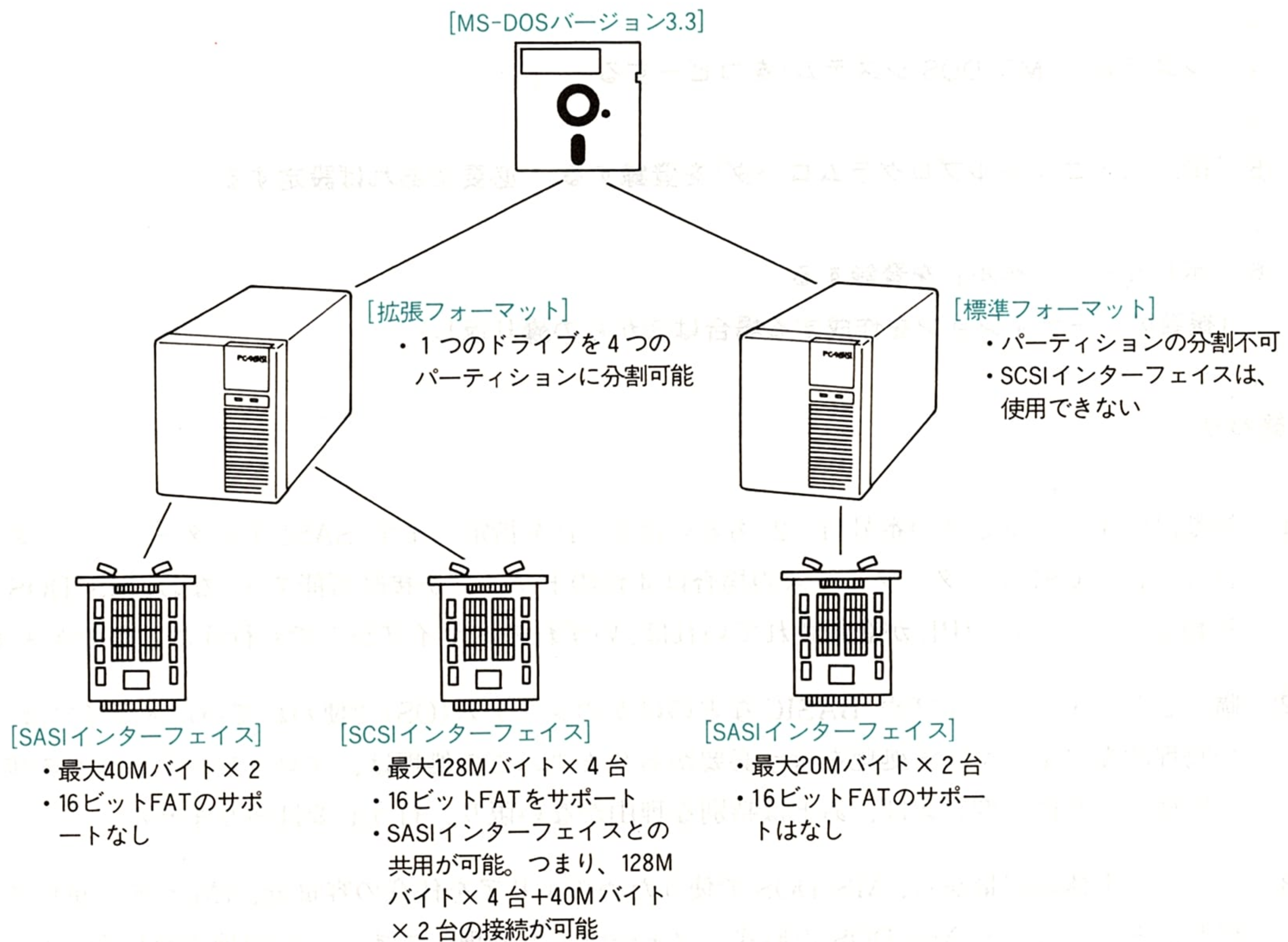



図4.7　MS-DOSバージョン3.3における接続可能なハードディスク

*注意：16ビットFATが使われている場合、12ビットFATに対応しているデバイスドライバが使われたりすると、ファイルが破壊されることもありうる。また、MS-DOSのバージョン2.*x*や3.1を起動した場合は、16ビットFATのハードディスクは使うことができない。

拡張フォーマットによる初期設定

では実際に、拡張フォーマットによるハードディスクの初期設定を行ってみましょう。拡張フォーマットの場合、初期設定は次のような手順で行います。

```
FORMAT コマンドを起動
 ↓
 ① ハードディスクの「装置番号」(ドライブ番号 1～2、あるいは 1～4)を指定する
 ↓
 ②「初期化」(ドライブ全体の物理的なフォーマット処理)を行う
 ↓
┌→③「領域確保」(MS-DOS が使用するエリアの論理的なフォーマット処理)を行う
│ ↓
│ ④「システム」(MS-DOS システム)をコピーする              ┐
│ ↓                                                      │
│ ⑤「IPL」(イニシャルプログラムローダ)を登録する             ├ 必要であれば設定する
│ ↓                                                      │
└─⑥「ボリュームラベル」を登録する。                         ┘
   (複数のパーティションを作成する場合は③からの繰り返し)
 ↓
終わり
```

①：初期設定するドライブの番号(1～2、あるいは 1～4)を指定します。SASI インターフェイスの場合は 2 台、SCSI インターフェイスの場合は 4 台のドライブが接続可能です。なお、MS-DOS の起動は、システムと IPL が登録されていれば、いずれのドライブからでも行うことができます。

②：購入したままのドライブや、BASIC などのほかのシステム(OS)で使われていたドライブは、この物理的なフォーマット処理を行う必要があります。この処理は、フロッピーディスクの場合と同様、最初に一度行えば、あとは特別な理由がない限り、行う必要はありません。

③：ドライブ全体の容量から、MS-DOS で使うためのエリアを任意の容量分、1M バイト単位で確保し、そのエリアを MS-DOS の形式でフォーマット処理します。この領域の確保が、1 つのパーティションを作成することになります。1 つのドライブを 2 つのパーティションに分けて使う場合には、全体の容量を適当に配分して、この領域確保の処理を 2 回行えばよいわけです。それぞれのパーティションに、システムや IPL を登録する場合は、③～⑥の作業を 2 回繰り返すことになります。

④⑤：ハードディスクから MS-DOS を起動するためには、この④⑤の操作により、MS-DOS システムと、その起動のためのプログラム(IPL)を登録しておく必要があります。起動は任意のドライブから行うことができます。

⑥：ボリュームラベルが必要なら、ここで付けることができます。ただしボリュームラベルは、「LABEL コマンド」を使えば、あとからでも付けたり／変更したり／削除することができます。

では、40M バイトのハードディスクをフォーマット処理(初期化)した後、20M バイトずつの 2 つのパーティションを作成します(つまり、20M バイトずつの 2 つの領域を確保します)。そして、パーティション①に MS-DOS のバージョン 3.3 のシステムをコピーし、パーティション②はシステムをコピーせず、ただのデータディスクとしましょう。つまり、次の図のように、1 つの 40M バイトのハードディスク上に、20M バイトの MS-DOS バージョン 3.3 のシステムディスクと、20M バイトのデータディスクを作成するわけです。

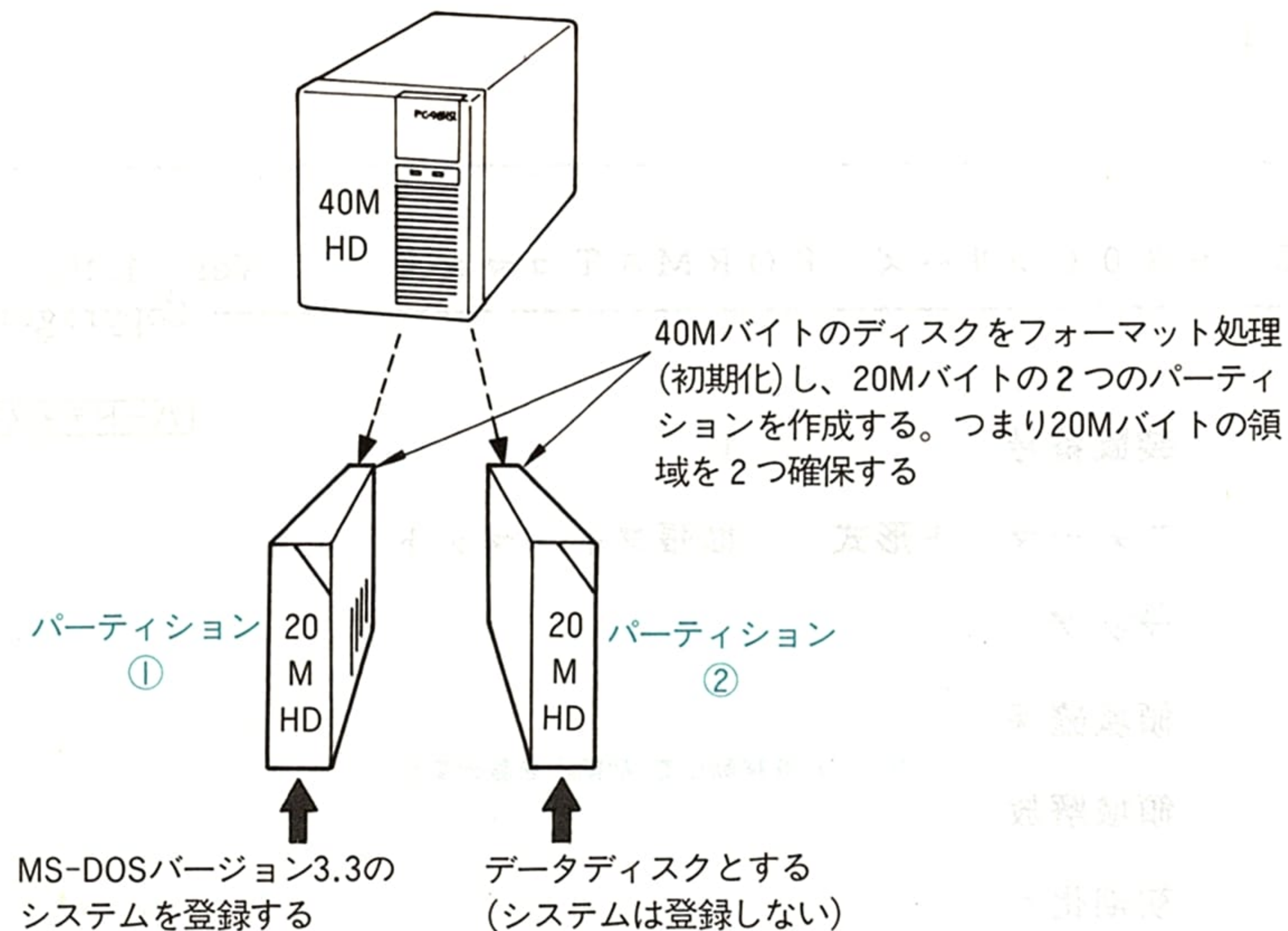



図 4.8　40M バイトのハードディスクを、20M バイト＋20M バイトの 2 つのパーティションに分割する概念図

では、その初期設定の作業を、まったくの最初から行う実行例を示しましょう。ドライブA：にMS-DOSのシステムディスクをセットして、接続したハードディスクの電源を必ずONにした後、MS-DOSを立ち上げてください。MS-DOSがすでに立ち上がっているあとから、ハードディスクの電源をONにしても、MS-DOSは、ハードディスクが接続されていることを認識しませんので注意が必要です。

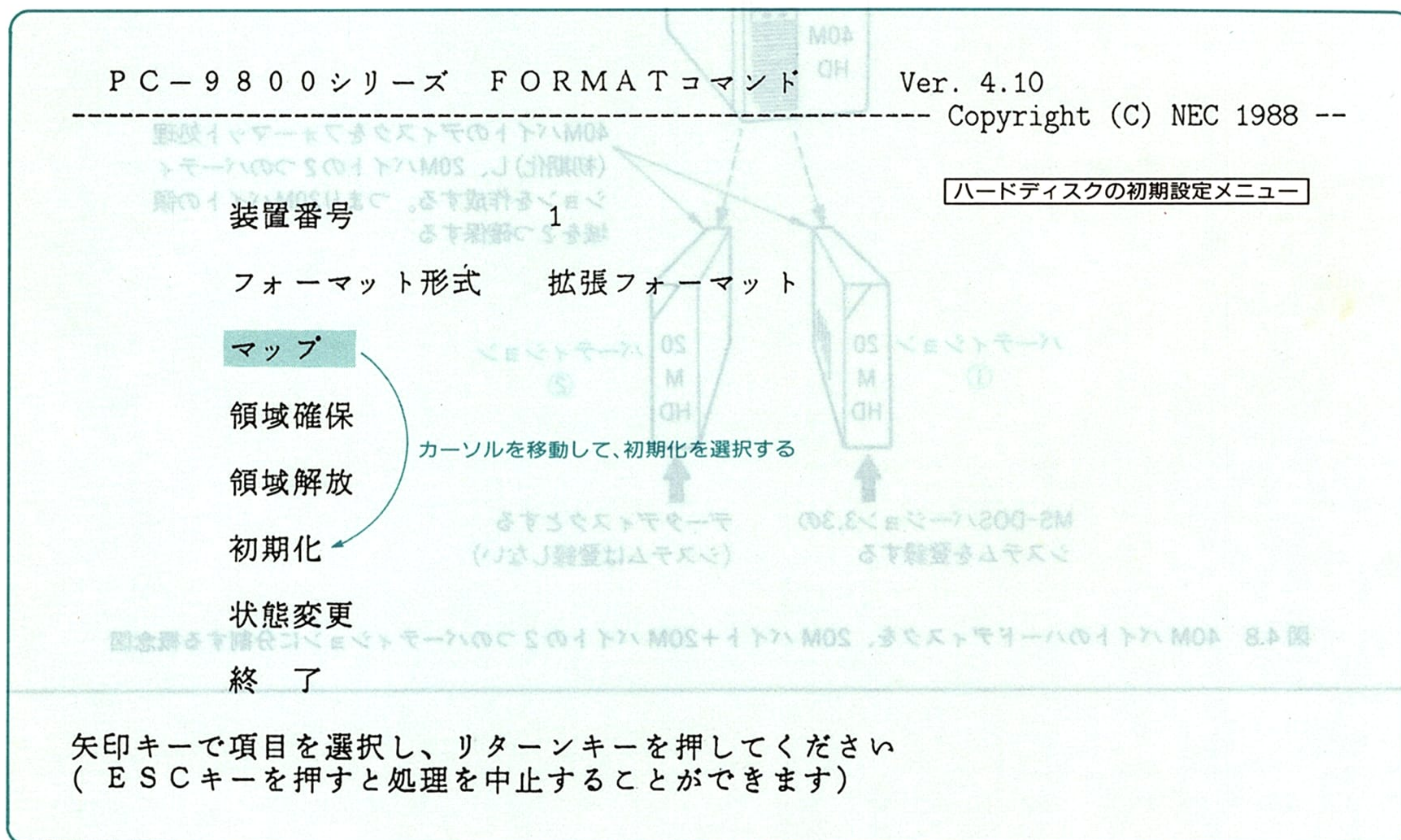
ハードディスクの初期設定は、フロッピーディスクと同じFORMATコマンドで行いますので、そのプログラムファイル「FORMAT.EXE」が必要です。確認しておいてください。なお、MS-DOSバージョン2.*x*に付属の「FORMAT.COM」では、拡張フォーマットの処理は行えません。

では始めましょう。ハードディスクの初期設定には、次のようにスイッチ「/H」を付けてFORMATコマンドを実行します。フロッピーディスクのフォーマット処理の場合に必要なドライブ指定は不用です。

```
A>FORMAT /H ⏎
```

スイッチ「/H」によって、フロッピーディスクのフォーマット処理とは違った、ハードディスク専用の初期設定モードにはいり、次のような画面が表示されます。実際の操作については、画面のなかで示しましょう。

```
  PC－9800シリーズ　FORMATコマンド    Ver. 4.10
------------------------------------------------------------ Copyright (C) NEC 1988 --

                                                 ハードディスクの初期設定メニュー
      装置番号            1

      フォーマット形式      拡張フォーマット

      マップ

      領域確保

      領域解放          カーソルを移動して、初期化を選択する

      初期化

      状態変更

      終　了

 矢印キーで項目を選択し、リターンキーを押してください
 （ＥＳＣキーを押すと処理を中止することができます）
```

⬇

装置全体を初期化します　フォーマットの形式を指定してください
（ＥＳＣキーを押すと処理を中止し、前画面に戻ることができます）

拡張フォーマット　標準フォーマット ……………拡張フォーマットを選択する

装置全体を初期化します　よろしいですか
（ＥＳＣキーを押すと処理を中止し、前画面に戻ることができます）

はい　いいえ ……………ハードディスクの物理的な初期化（フォーマット）を行う

装置の初期化を終了しました　どれかキーを押してください
残り　０メガバイトです
0 10 20 30 40 50 60 70 80 90 100 (%)

……………ハードディスクの初期化が終了（40Mバイトで約7分かかる）

初期設定メニューにもどるので、マップを選択しハードディスクの状況を確認してみる

ＰＣ－９８００シリーズ　ＦＯＲＭＡＴコマンド　Ver. 4.10
-- Copyright (C) NEC 1988 --

装置番号　1

フォーマット形式　拡張フォーマット

システム名	状　態	FROM		TO(ｼﾘﾝﾀﾞ)	ｻｲｽﾞ	BOOT
未使用領域		0001	～	0613	040	

……………物理的なフォーマットをしただけでは、まだ使用できないので「未使用領域」になっている

初期設定メニューにもどるので、領域確保を選択する

確保容量　　020 ＭＢ

領域確保（MS-DOSのディスクとして使用可能にする）のためのメニュー

先頭シリンダ　　0001

システム　　転送する

ボリュームラベル

実　行

確保する容量は何メガバイトですか　HELPキーを押すとマップを表示します
確保可能な容量は １ ～ 040 ＭＢです
（ ＥＳＣキーを押すと処理を中止し、前画面に戻ることができます）
確保容量＝20 ……………パーティション①（図4.8参照）のための領域を指定する

先頭シリンダ　　0001 ……………図4.12を参照

システム　　転送する

ボリュームラベル

実　行

領域の先頭シリンダ番号を指定してください　HELPキーを押すとマップを表示します
リターンキーのみ入力した場合は、空き領域の先頭から確保します
（ ＥＳＣキーを押すと処理を中止し、前画面に戻ることができます）
シリンダ番号＝ ……………最初のシリンダから確保するので、キーのみでよい

システムを転送しますか
矢印キー（ ↑・ ↓・ ←・ →）で項目を選択し、リターンキーを押してください
（ ＥＳＣキーを押すと処理を中止し、前画面に戻ることができます）
転送する　転送しない ……………パーティション①にはシステムを登録する

ボリュームラベルを入力してください
漢字<全角>は５文字、英数字<半角>は１１文字まで、必要なければリターンキー
（ ＥＳＣキーを押すと処理を中止し、前画面に戻ることができます）
ボリュームラベル＝MS-DOS V3_3 ……………ボリュームラベルを設定する（あとからLABELコマンドで付加／変更／削除することも可能）

領域の確保を行います　準備はよろしいですか
（はい：初期化する　いいえ：初期化しない）
（ESCキーを押すと処理を中止し、前画面に戻ることができます）
はい　いいえ ……………確認のメッセージが表示される
（このあと領域確保の処理が開始される）

⬇

システムを転送しました ……………領域確保が終了すると（40Mバイトで約8分かかる）、
どれかキーを押してください　カレントドライブ（ドライブA：）からシステムが転送される

⬇

PC－9800シリーズ　FORMATコマンド　Ver. 4.10
-- Copyright (C) NEC 1988 --

装置番号　1

フォーマット形式　拡張フォーマット

マップ

領域確保 ……………パーティション②（20Mバイト）の領域を確保し、MS-DOSで使用可能にする。操作は前述の場合と同じ。違うところのみを以下の図に示す

領域解放

⬇ 確保する容量として20Mバイトを指定する

先頭シリンダ　0301 ……………空き領域の先頭のシリンダが"301"になっている。つまり、0～300まではパーティション①に使われたことになる

システム　転送する

ボリュームラベル

実　行

領域の先頭シリンダ番号を指定してください　HELPキーを押すとマップを表示します
リターンキーのみ入力した場合は、空き領域の先頭から確保します
（ESCキーを押すと処理を中止し、前画面に戻ることができます）
シリンダ番号＝⏎ ……………空き領域の先頭（すなわち"301"）からでよいので、⏎を入力

⬇

図 4.9　拡張フォーマットにおける 40M バイトのハードディスクの設定例

以上の作業で、1台の 40M バイトのハードディスクを、20M＋20M バイトの2つのパーティション —— 20M バイトの MS-DOS バージョン 3.3 のシステムディスクと、20M バイトのデータディスク —— に分割した初期設定が終わりました。ただしハードディスクは、このままの状態では動作しません。必ず MS-DOS を再起動する必要があります。

では、ドライブ A：のシステムディスクを取り出した後、リセットボタンを押し、設定したハードディスクから MS-DOS を立ち上げてみましょう(PC-9801 では、フロッピーディスク／ハードディスクのどちらからでも MS-DOS の立ち上げを可能にするには、SWITCH コマンドの[BOOT]の設定を「標準」(STD)にしておきます)。

（リセットボタンを押す）　ドライブＡ：のディスクは取り出しておく

```
NEC PC-9800 ｼﾘｰｽﾞ
固定ディスク起動メニュープログラム ﾊﾞｰｼﾞｮﾝ 1.05
Copyright (C) 1985,1986 NEC Corpor

      固定ディスクドライブ＃1

  処　理: 起動する　自動起動に設定する　次のドライブ

  領域①: MS-DOS 3.30
      ②: MS-DOS 3.30
      ③:
      ④:
      ⑤:
      ⑥:
      ⑦:
      ⑧:

説明: カーソル移動キー＝処理／領域の選択　リターンキー＝実行
```

パーティション①(20Mバイト)

パーティション②(20Mバイト)

起動する領域（パーティション）を選択する。ただしここでは、パーティション②はデータディスク（システムがはいっていない）に設定してあるため選択できない

↵キーを入力すると、パーティション①から起動する

```
NEC PC-9800 Series Personal Computer

ﾏｲｸﾛｿﾌﾄ MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30
Copyright 1981,88 Microsoft Corp. / NEC Corporation

Command ﾊﾞｰｼﾞｮﾝ 3.30
現在の日付は 1989-09-16 (土) です.
日付を入力してください:↵
現在の時刻は 23:43:43.00 です.
時刻を入力してください:↵

A>DIR↵

 ドライブ A: のボリュームラベルは MS-DOS V3_3
 ディレクトリは A:¥

COMMAND  COM    24931  88-07-13   0:00
        1 個のファイルがあります
 20086784 バイトが使用可能です
```

パーティション①のMS-DOSバージョン3.3が起動した

パーティション①（ドライブＡ：）の内容を確認する

フォーマットプログラムで設定したボリュームラベル

システムを転送したので、COMMAND.COMがコピーされている

設定したとおり容量は20Mバイト

```
A>C:¥CHKDSK A:⏎
ディスク MS-DOS V3_3 は 1989-9-16 23:23 に作成されました.

 20217856 バイト : 全ディスク容量
    98304 バイト : 3 個のシステムファイル
    32768 バイト : 1 個のユーザーファイル
 20086784 バイト : 使用可能ディスク容量

   655360 バイト : 全メモリ
   564640 バイト : 使用可能メモリ

A>C:¥CHKDSK B:⏎
ディスク DATA        は 1989-9-16 23:27 に作成されました.

 20217856 バイト : 全ディスク容量
        0 バイト : 1 個のシステムファイル
 20217856 バイト : 使用可能ディスク容量

   655360 バイト : 全メモリ
   564640 バイト : 使用可能メモリ

A>
```

コンピュータ本体内蔵のフロッピーディスクドライブが"C："になる
CHKDSKコマンドで、ディスクの容量を確認する
フォーマットプログラムで設定したボリュームラベル
MSDOS.SYS、IO.SYS、ボリュームラベルの3つのファイル
COMMAND.COMファイル
パーティション②(ドライブB：)の内容を、CHKDSKコマンドで確認する
フォーマットプログラムで設定したボリュームラベル
システムを転送していないため、すべて空き容量になっている

図 4.10　初期設定したハードディスクのパーティション①から MS-DOS を立ち上げる

拡張フォーマットで初期設定されたハードディスクを立ち上げると、マシンのメモリチェックが行われた後、177 ページの上に示したような「固定ディスク起動メニュー」画面が表示されます(PC-9800 シリーズの場合)。これについてくわしくは次節で解説しますが、とりあえず「領域①」(つまりパーティション①)を選択して⏎キーを押してみましょう(パーティション②は、MS-DOS システムが登録されていないデータディスクであり、選択しても MS-DOS を起動することはできない)。

パーティション①に登録してある MS-DOS システム(バージョン 3.3)が立ち上がりました。パーティション②の領域は、物理的には同一ドライブですが、MS-DOS 上では「ドライブ B：」に割り当てられています。これがパーティションによる分割の一例です。

さてこのように、ハードディスクから MS-DOS が起動しました。COMMAND.COM が 1 つだけコピーされていますが、これは 170 ページの④にあたる「システムの転送」作業で、MS-DOS システムとともにコピーされたものです。

これで 40M バイトのハードディスクが、拡張フォーマットにより完全に使える状態になりました。このあとは、各ユーザーが、この 20M+20M バイトという大容量のディスクを、いかにうまく利用するかという問題になります。

4.3 パーティションと任意のパーティションからの起動

パーティション(Partition)とは、「分割」とか「仕切る」とかいう意味であり、拡張フォーマットを行うことによって、1台のハードディスクを複数のパーティションに分割することができます。このパーティション分割の機能を利用すれば、ほかの複数のOSを、1台のディスク上に共存させたり、大容量のディスクを適当な大きさの複数のMS-DOSのディスクに分割したりすることができます。本節では、大容量のハードディスクをうまく活用するために必要な機能である「パーティション」について解説しましょう。

■ パーティションとは

拡張フォーマットでは、パーティションに分割することによって、1台のドライブを、見かけ上、複数のドライブに分割することが可能です。つまり、複数の〝ハードディスクドライブ〟になるわけですから、その1台1台を、「何にどう使おうと勝手」であり、ほかの〝ドライブ〟には影響を与えません。

また、「何にどう使おうと勝手」であると同時に、それぞれのドライブの〝電源〟を入れたり切ったりするのも勝手です。つまり、それぞれのドライブを「生かす／生かさない」を自由に選択できるわけです(次項のアクティブ／スリープ)。

パーティションとは、このような単純な機能です。見かけ上、それぞれ独立しているドライブを、どのようにうまく利用するかは、ユーザーの使い方しだいです。たとえば、ディスクのメモリ容量を有効利用するため、後述するクラスタサイズが小さくなるように、適当な大きさのパーティションに分割するとか、あるドライブを、MS-DOSの別のバージョンのシステムディスクにするとか(ただし、MS-DOSバージョン2.*x*は、拡張フォーマット上のハードディスクから起動することはできない)、あるいはMS-DOS以外の別のOS用のドライブにするのもよいでしょう。

各ドライブは、それぞれのシステムやIPLが登録されていれば、起動の際の「起動メニュー」により、任意のドライブから自由に起動することができます。また、不用なドライブであれば、そのドライブの〝電源〟をOFFにしておくこともできます。もちろんここでの〝ドライブ〟とはパーティションのことですが、このことを次ページの図で示しましょう。

このように、それぞれのパーティションは、〝それぞれ独立したドライブ〟ですから、MS-DOSが起動した時点で、電源が〝ON〟(**アクティブ**)のパーティションには、それぞれのドライブ名が割り当てられます。電源が〝OFF〟(**スリープ**)のパーティションは、ドライブが接続されていないとみなされます。

40M
HD

D:
E:

上の40Mバイトのハードディスクを下の5つのパーティションに分割した状態の概念図。

これらのそれぞれが「パーティション」。それぞれが独立したドライブと考えればよい。

パーティション①
パーティション②
パーティション③
パーティション④
パーティション⑤

10M　10M　5M　5M　10M

ここに記したドライブ名は、このパーティション①から、MS-DOSを起動した場合の一例

A:　B:　C:

(MS-DOS バージョン3.3のシステムディスク)
(データディスク)
(MS-DOS バージョン3.3のシステムディスク)
(データディスク)
(他のOSのシステムディスク)

それぞれのパーティションの利用の一例

“電源”ON（つまりアクティブの状態）
“電源”OFF（つまりスリープの状態）

それぞれの“ドライブ”の“電源”のON/OFFが、パーティションのアクティブ/スリープの選択に相当する。現在、パーティション①②④の3つのパーティションが「アクティブ」である。同時にアクティブにできるパーティションは、最大4つである。

図 4.11　パーティションの概念とその利用法

図 4.11 では、5つのパーティションのうち、3つの“電源”が ON になっていますので、パーティション①から MS-DOS を立ち上げた場合は、図のようなドライブ名の割り当てになります。つまり、それぞれのパーティションは、ドライブ名の違いによって区別されるわけです。

1つの「パーティション」の実体は、実際のディスク上の、ある範囲の連続した**シリンダ**に対応しています。前節の図 4.9（176 ページ）の「領域確保」の作業画面では、40M バイトのハードディスク内部のシリンダを、No.1〜300 の範囲をパーティション①に、No.301〜600 の範囲をパーティション②に割り当てています。それを次の図で示しましょう。

なお「シリンダ」とは、一般的には、ディスク装置の磁気記録円盤(大容量のハードディスクでは、普通、複数枚が重ねられている)の、中心からの距離が同じ位置にある、トラック全体を指す呼び名です。

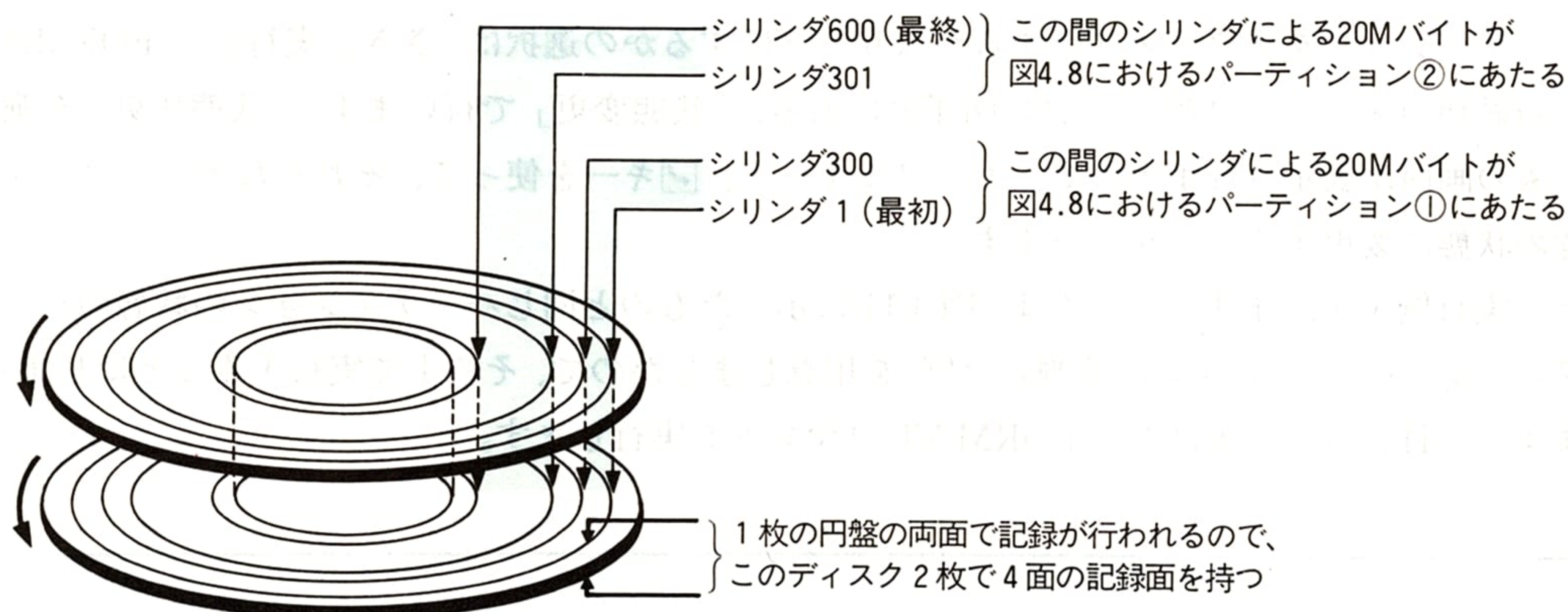



図 4.12　シリンダの概念

それぞれのパーティションは、このように連続したシリンダを割り当てることによって実現されています。しかし、私たちがこのシリンダ番号を使って領域を設定するのはたいへんです。そこで、領域を①から順番に確保していけば、それぞれのパーティションの容量を指定するだけで(10M バイトとか 20M バイトとか)、シリンダ番号は自動的に設定されるようになっています(図 4.9 参照)。

実際のパーティションを設定する操作は、図 4.9 で行いましたが、複数のパーティションに分割するための特別な操作法があるわけではありません。拡張フォーマットを行う際の、「領域確保」の操作そのものが、ある 1 つのパーティションを確保するための操作なのです。つまり、拡張フォーマットの「領域確保」とは、そもそもパーティションを設定することなのです。その〝特別な例〟として、1 つのパーティションに、そのディスクの全容量を割り当てた場合が、複数のパーティションに分割しない例なのです。

さて、パーティションの分割とは、実はディスクのシリンダの確保であることがわかりましたが、その分割の数には限度があります。分割可能な最大数は 8 つ(PC-9800 シリーズの場合)ですが、同時にアクセス可能なパーティション(〝アクティブ〟のパーティション)は最大 4 つです。つまり、8 つのパーティションに分割できるものの、同時に〝電源〟を ON にして使うことができるドライブの数は 4 つまでです。その〝電源〟の ON/OFF の選択が、次項で解説するパーティションの「アクティブ／スリープ」の指定なのです。

■ パーティションの「アクティブ」と「スリープ」

拡張フォーマットにおいては、1台のハードディスクを4つ以上のパーティションに分割することができますが、それらのパーティションの、どれを生かしてどれを眠らせておくかを、あらかじめ指定しておくことができます。使用可能な生きているパーティションを**アクティブ**、使用できない眠っているパーティションを**スリープ**と呼びます。

パーティションをアクティブにするかスリープにするかの選択は、さきに実行したFORMATコマンドの最初のメニュー(172ページの図4.9)にある、「状態変更」で行います。「状態変更」を選択すると、次の画面が表示されますので、カーソルキーと、⏎キーを使って、それぞれのパーティションを任意の状態に変更することができます。

その実行例を示します。ここでは、図4.11に示したものと同じパーティションの状態のハードディスク(5つのパーティションに分割した例)を用意しましたので、その上で実行することにしましょう。

まず、「/H」スイッチ付きのFORMATコマンドを実行します。

```
  ＰＣ－９８００シリーズ　ＦＯＲＭＡＴコマンド　　　Ver. 4.10
---------------------------------------------------- Copyright (C) NEC 1988 --

        装置番号            1
        フォーマット形式    拡張フォーマット

     システム名      状　態       FROM       TO(シリンダ)   サイズ   BOOT

  ① MS-DOS 3.30    アクティブ    0001   ～   0150         010      可
  ② DATA DISK1     アクティブ    0151   ～   0300         010      不可
  ③ 1TARO V4.2      スリープ     0301   ～   0375         005      可
  ④ DATA DISK2     アクティブ    0376   ～   0450         005      不可
  ⑤  他のＯＳ                    0451   ～   0600         010
```

FORMATコマンドの初期設定メニューから「状態変更」を選択する

まずこの「状態」を選択する

MS-DOSシステムを転送してあるパーティション

起動メニューに表示されるシステム名も、この「システム名」を選択して、任意の名前に変更することができる。なお、ここに表示される名前は「ボリュームラベル」ではないことに注意

⏎キーを押すことにより、アクティブ／スリープの状態を変更することができる

各パーティションの容量(Mバイト)

ESC によって処理を終了し、初期設定メニューにもどる

図4.13　各パーティションのアクティブ／スリープを指定する

この実行例で設定した状態は、図4.11の状況を実際に再現するためのもので、パーティション①②④がアクティブであり、③がスリープです(⑤はBASICの領域なのでアクティブ／スリープなどはありません)。ではFORMATコマンドを終了した後、MS-DOSを再起動してみましょう。

リセットボタンを押すと、MS-DOSの起動前に、次ページのような「固定ディスク起動メニュー」が表示されます。

```
（リセットボタンを押す）

NEC PC-9800 ｼﾘｰｽﾞ
固定ディスク起動メニュープログラム ﾊﾞｰｼﾞｮﾝ 1.05
Copyright (C) 1985,1986 NEC Corpor

          固定ディスクドライブ＃1

  処　理: 起動する　自動起動に設定する　次のドライブ

  領域①: MS-DOS 3.30 ……システムディスク（10Mバイト）。アクティブ
      ②: DATA DISK1  ……データディスク（10Mバイト）。アクティブ
      ③: 1TARO V4.2  ……システムディスク（5Mバイト）。スリープ
      ④: DATA DISK2  ……データディスク（5Mバイト）。アクティブ
      ⑤: N88-BASIC   ……DISK BASIC（10Mバイト）
      ⑥:
      ⑦:
      ⑧:

  説明: カーソル移動キー＝処理／領域の選択　リターンキー＝実行
```

さきの「システム名」であり、MS-DOSの「ボリュームラベル」ではないことに注意

システムディスクとアクティブ／スリープの関係から、現在起動できるのは①と⑤の領域のみ

図 4.14　パーティションのアクティブ／スリープを設定したディスクを起動する

パーティション③は、ブート可能なシステムディスクですが、状態がスリープであるため、現在は起動できません。では、パーティション①（MS-DOS バージョン 3.3 のシステムディスク）を起動して、接続されている全ドライブの状態をチェックしてみましょう。

起動メニューの「領域①」を選択して⏎することにより、パーティション①の MS-DOS システムが起動します。起動後の CHKDSK コマンドの実行例を示しましょう。

```
本体のドライブD：にMS-DOSのシステムディスクを入れておく
A>D:¥CHKDSK A:⏎ ……ドライブA：の内容を確認する
ディスク MS-DOS V3_3 は 1989-9-17 20:49 に作成されました．

 10100736 バイト : 全ディスク容量
    98304 バイト : 3 個のシステムファイル
    28672 バイト : 1 個のユーザーファイル
  9973760 バイト : 使用可能ディスク容量
        :
A>D:¥CHKDSK B:⏎ ……ドライブB：の内容を確認する
ディスク DATA1       は 1989-9-17 21:02 に作成されました．

 10100736 バイト : 全ディスク容量
        0 バイト : 1 個のシステムファイル
 10100736 バイト : 使用可能ディスク容量
        :
```

容量は10Mバイトでシステムが転送されている

容量は10Mバイトでデータディスクになっている

```
A>D:¥CHKDSK C:⏎ ……………ドライブC：の内容を確認する
ディスク DATA2        は 1989-9-17 21:10 に作成されました.

  5042176 バイト : 全ディスク容量
        0 バイト : 1 個のシステムファイル     容量は5Mバイトで
  5042176 バイト : 使用可能ディスク容量       データディスクになっている
          ⋮
A>                              ＊CHKDSKコマンドのメッセージは省略してある
```

図 4.15　パーティション①の MS-DOS を起動し、各ドライブをチェックする

CHKDSK コマンドで各ドライブをチェックした結果、180 ページの図 4.11 に示したとおりのドライブに割り当てられていることがわかりました。

なお、「固定ディスク起動メニュー」の操作により、リセットボタンや電源を ON にするだけで、あとは手放しで目的のパーティションを起動することもできます(その場合、SWITCH コマンドの「BOOT 装置」の指定を「固定ディスク#n」に設定しておくこと。n はハードディスクの番号)。MS-DOS の起動がかかり、もしそのパーティションに自動スタート・バッチファイルが存在していれば、それが自動実行されます。結局、フロッピーディスクと同様に、任意のアプリケーションプログラムを自動スタートすることができるわけです。

それを設定するための実行例を示しましょう。パーティション①を自動起動に設定します。設定は、図 4.14 で示した「固定ディスク起動メニュー」で行います。

```
NEC PC-9800 ｼﾘｰｽﾞ
固定ディスク起動メニュープログラム ﾊﾞｰｼﾞｮﾝ 1.05
Copyright (C) 1985,1986 NEC Corpor

            固定ディスクドライブ＃1
                        ⓐ
  処　理: 起動する  自動起動を解除する  次のドライブ
                        ⓒ                ⓑ
  領域①: MS-DOS 3.30      ＊
      ②: DATA DISK1
      ③: 1TARO V4.2
      ④: DATA DISK2          ⓐ「自動起動を設定する」を選択する
      ⑤: N88-BASIC           ⓑ領域①を選んで⏎キー。
      ⑥:                      領域の最後に“＊”マークがつく。
      ⑦:                      メニューが「自動起動を解除する」
      ⑧:                      に変更される
                              ⓒ「起動する」を選んで⏎キー

説明: ＊は自動起動に設定された領域です
```

⬇ MS-DOSを起動後、SWITCHコマンドを実行して、「BOOT装置」を「固定ディスク＃n」に設定しておくこと

```
  ＢＯＯＴ装置            標準 ……………MS-DOSを起動するドライブを
                                    固定ディスク＃1に設定する
  数値データプロセッサ２  無

  終 了

システムを起動するディスク装置を指定してください
標準を指定するとシステムの入っている装置から起動します
（ＥＳＣキーを押すと処理を中止することができます）
標準 1MBFD 640KBFD 固定ディスク#1 固定ディスク#2 SCSI固定ディスク
```

図 4.16a 任意のパーティションを自動起動に設定する

この作業で、パーティション①の MS-DOS が自動起動に設定されました。では、このパーティションにワープロソフト一太郎をコピーし、その自動スタート・バッチファイルを登録した後、リセットボタンを押してみましょう。

```
(リセットボタンを押す)
NEC PC-9800 Series Personal Computer                固定ディスク起動メニューが表
                                                    示されずにパーティション①が立
ﾏｲｸﾛｿﾌﾄ MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30                          ち上がった。
Copyright 1981,88 Microsoft Corp. / NEC Corporation 他のパーティションから起動した
                                                    いときは、SWITCHコマンドで
自由文変換システム ＡＴＯＫ６ ver 1.2               BOOT装置を［標準］にもどし
Copyright 1986,87 (株)ジャストシステム              再度リセットすればよい

Command ﾊﾞｰｼﾞｮﾝ 3.30

A>JXW ……………一太郎を起動する自動スタート・バッチファイルが実行される
```

⬇

```
[1]            [2]        《一太郎》        1頁  1行  1列
[1] ■------10--------20--------30--------40--------50--------60--------70-+--
 |                  一太郎が起動した
```

図 4.16b ワープロソフトを自動スタートする

このように、「固定ディスク起動メニュー」の設定により、リセットボタンや電源 ON の操作だけで、ハードディスク上の任意のパーティションの任意のアプリケーションプログラムを自動的に立ち上げることができました。

4.4　12ビットFAT／16ビットFAT　その違い

MS-DOSバージョン2.*x*から3.*x*へのバージョンアップにおける重要な変更点の1つに「FAT」があります。『大容量のハードディスクに対応するために、16ビットFATをサポートした』といわれていますが、FATのビット数を、従来の12ビットから16ビットに拡張すると、何がどのように変わるのでしょう。

また、ハードディスクでは、1つファイルを作るごとに、8Kバイトとか、16Kバイトとかの単位で、ディスク上のメモリが消費されていきます。そこには無駄に捨てられる部分も多く含まれ、ディスク上のメモリの使用効率の面で問題があります。この問題の解決にも、FATのビット数が関係しているのです。

本節では、これらの問題について簡単に解説しましょう(さらにくわしくは、「応用MS-DOS」で解説している)。

■ クラスタ

さて、一般のディスク装置において、ファイルのデータがディスクに書き込まれる際の状況について考えてみましょう。ファイルのデータサイズ(大きさ。バイト数)はさまざまですが、ディスクへの書き込みは、1バイト単位の可変長で行われるわけではなく、ある大きさのブロック単位で行われます。その物理的な単位が、ディスクフォーマットにおける**1セクタ**です。ディスク装置は、どのようなファイル、どのようなデータでも、必ず「1セクタ」を読み／書きの単位としています。この1セクタの大きさは、フロッピーディスク／ハードディスクとも、一般的には1024バイト(1Kバイト)のものが多いようです。

しかし「セクタ」は、ディスク装置における物理的な読み／書きのデータブロックであり、MS-DOSのファイル管理上のデータブロックの単位ではありません。MS-DOSのファイル管理上のデータブロック ―― つまり論理的なデータの単位が**クラスタ**なのです。

MS-DOSは、ディスク上のすべてのファイルのデータを、クラスタ単位で管理しています。一般に使われている1Mバイトタイプのフロッピーディスク(3.5インチや5インチの2HDタイプなど)におけるクラスタのサイズは、1024バイト(1Kバイト)です。つまり、1Mバイトタイプのフロッピーディスクにおけるクラスタのサイズは、1セクタと同じです。しかしこのクラスタのサイズは、ディスクの容量やフォーマットの形式などによって異なり、たとえば、標準フォーマットのハードディスクの場合は、ディスクの容量にかかわらず、すべて8Kバイトです。では、1Kバイトのクラスタと、8Kバイトのクラスタとは何が違うのでしょう。そこがポイントです。

MS-DOS は、ディスク上のファイルを、クラスタ単位で管理しています。逆にいえば、クラスタサイズより小さいデータでも、最小限、クラスタサイズとして扱われます。具体的には、たとえ 1 バイトのデータであっても、そのデータのファイルを作成すれば、フロッピーディスクであれば 1K バイト、標準フォーマットのハードディスクであれば 8K バイトものディスク上のメモリが消費されるわけです。

このようなことは、各ファイルの最後のクラスタについても言えます。クラスタサイズでちょうど割り切れるようなファイルはまれですから、たいていのファイルの最後には〝あまり〞が出ます。これも極端な例ですが、8K バイト＋1 バイト（8193 バイト）のデータのファイルを作成すれば、フロッピーディスクであれば 9K バイト、同じく標準フォーマットのハードディスクであれば 16K バイトものディスク上のメモリが消費されます。

さらにいえば、内容がたった 1 バイトのファイルであっても、それを 10 個作成すれば、フロッピーディスクで 10K バイト、ハードディスクでは 80K バイトもの容量を消費するわけです。このことを次の図で示しましょう。

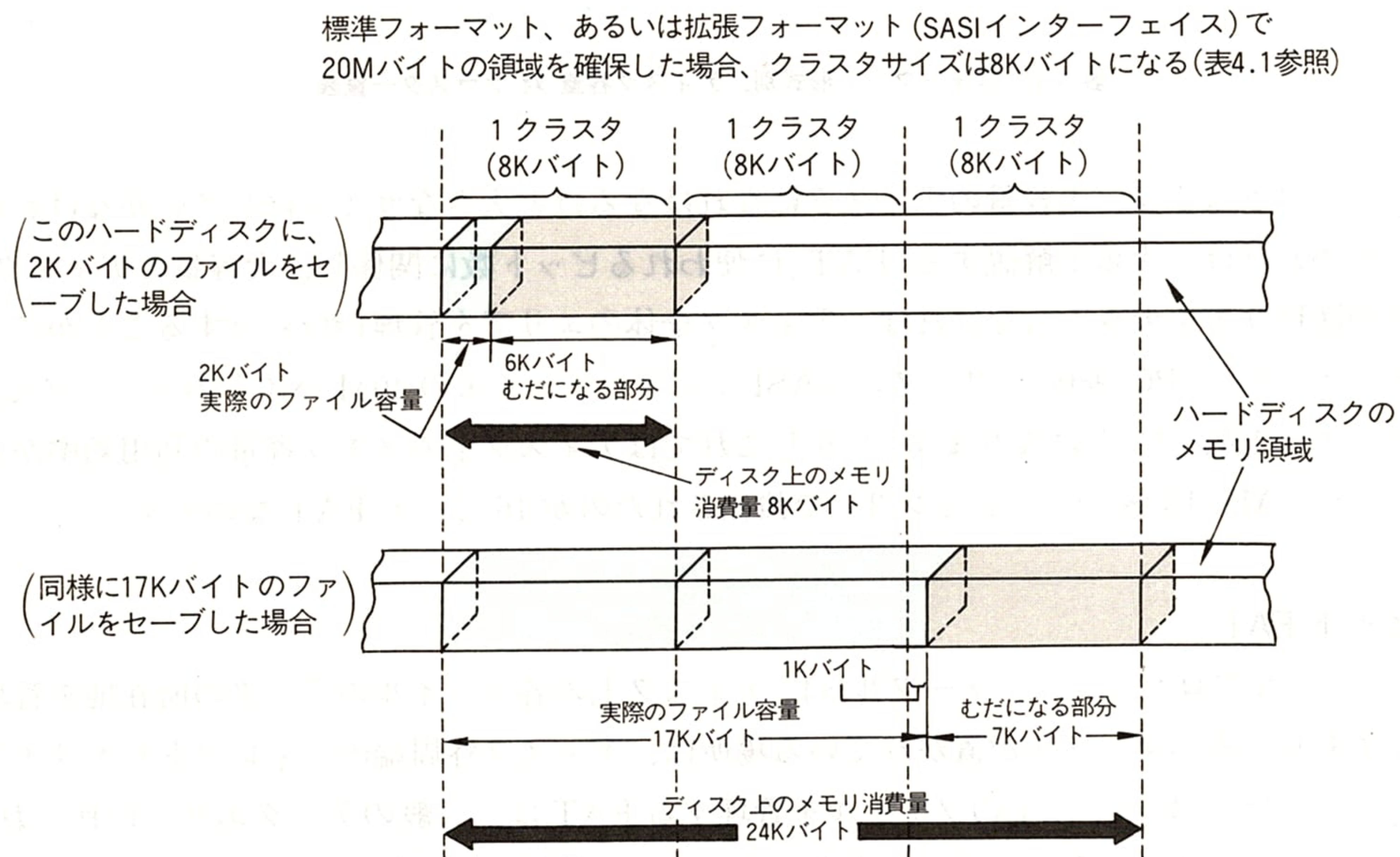



図 4.17　ファイル管理の単位―クラスタの概念

〈標準フォーマット〉

確保する容量	1Mバイト	5Mバイト	10Mバイト	15Mバイト	20Mバイト
クラスタサイズ	8Kバイト	8Kバイト	8Kバイト	8Kバイト	8Kバイト
FATのビット数	12ビット				

確保する容量によらず、
クラスタサイズは8Kバイトに固定

〈拡張フォーマット〉

●SASIインターフェイスの場合

確保する容量	1〜5Mバイト	6〜15Mバイト	16〜30Mバイト	31〜40Mバイト
クラスタサイズ	2Kバイト	4Kバイト	8Kバイト	16Kバイト
FATのビット数	12ビット			

●SCSIインターフェイスの場合

確保する容量	1〜5Mバイト	6〜10Mバイト	11〜64Mバイト	65〜128Mバイト
クラスタサイズ	2Kバイト	4Kバイト	2Kバイト	4Kバイト
FATのビット数	12ビット		16ビット	

SASIインターフェイスの
場合と同じ

表 4.1　フォーマット形式別、ディスク容量 対 クラスター覧表

さらにこのクラスタは、大容量のディスクになればなるほど大きなサイズにしていかなければなりません。そのわけは、次項で解説する「FAT」に使われるビット数に関係し、大容量のディスクでは、クラスタの区切り方を大きくしなければ、ディスク全体のエリアを管理(カバー)することができないからです。たとえば、PC-9800シリーズのSASIインターフェイスの40Mバイトのディスクでは、クラスタサイズは16Kバイトになります。しかしこれではディスク上のメモリ容量の利用効率があまりにも悪いため、MS-DOSのバージョン3.*x*で強化されたのが16ビットFATなのです。

■16ビットFAT

FAT(ファイルアロケーションテーブル)は、ディスク上の各ファイルのデータの所在地を管理するテーブルであり、そのテーブルが置かれている場所は、ディスク外周端のディレクトリエリアのさらに外側です(サブディレクトリ上のファイルを管理するFATは、一般のデータエリアに作られる)。ディスク上に書き込まれた各ファイルの中身のデータは、前項で解説した「クラスタ」という単位に分割され、そのそれぞれの所在をFATが管理しています。

MS-DOSバージョン2.*x*では、1つのクラスタを管理するFATは12ビットで構成されていましたが、バージョン3.*x*では、それが16ビットに拡張されました。大容量のハードディスクに対応するためです。ただし、12ビットをやめて全面的に16ビットに切り替えたわけではなく、「必要があれば16

ビットのFATも使える」という意味です。たとえばSCSIインターフェイスを使ったハードディスクの例では、領域確保が10Mバイト以下の場合は12ビットFATを使い、11Mバイト以上になると16ビットFATを使うといった具合です(表4.1参照)。

ここでFATの働きを、次の簡略化した模式図で示しましょう。2ビットFATと4ビットFATというものを考え、それで容量10Mバイトのハードディスクを管理することを想定します。

2ビットで表現できる数は、「00」「01」「10」「11」の4つですから、2ビットFATでは、最大4つの場所しか管理することはできません。では4ビットFATになるとどうなるでしょう。4ビットでは、「0000」「0001」「0010」～「1111」の16通りの数を表現できますので、最大16の場所を管理することができます。10Mバイトというのは、10240Kバイトのことですから、この様子は次の図のようになります。

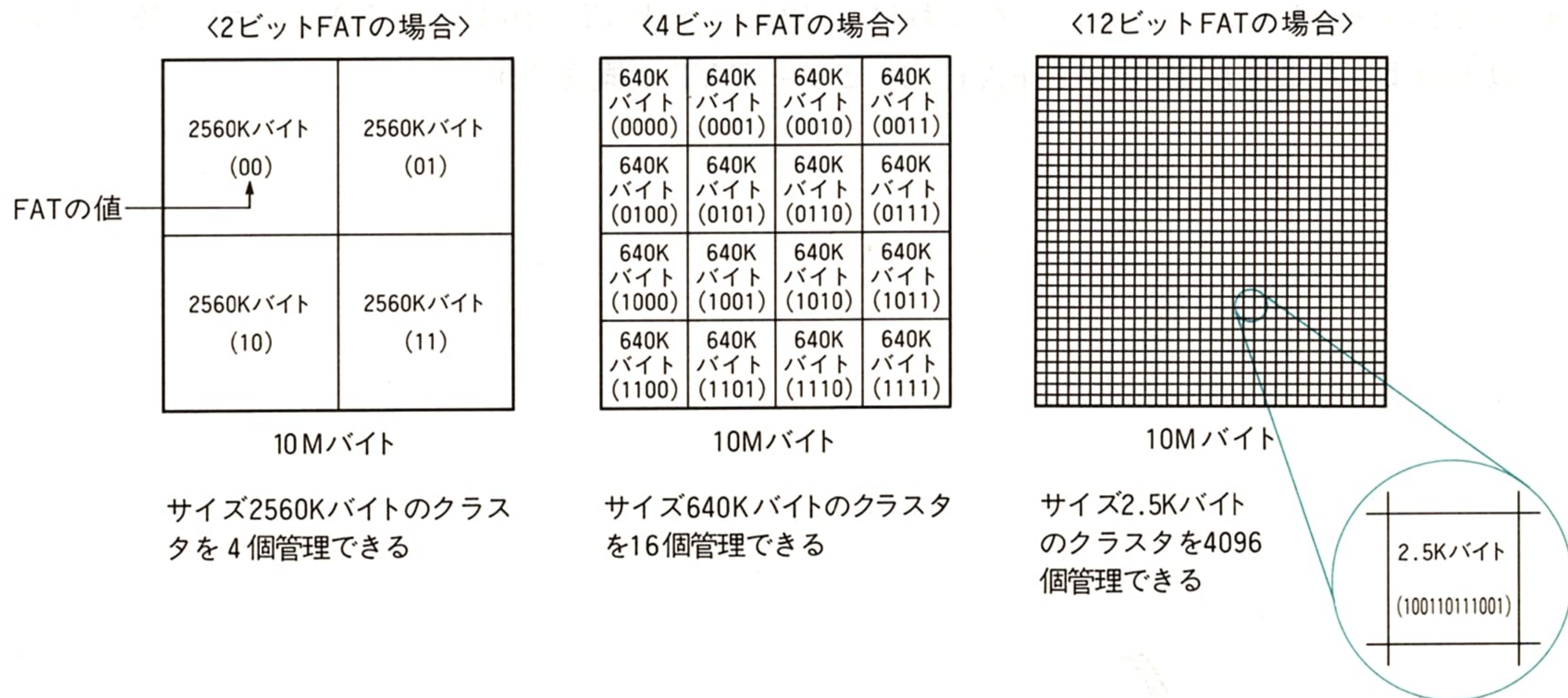



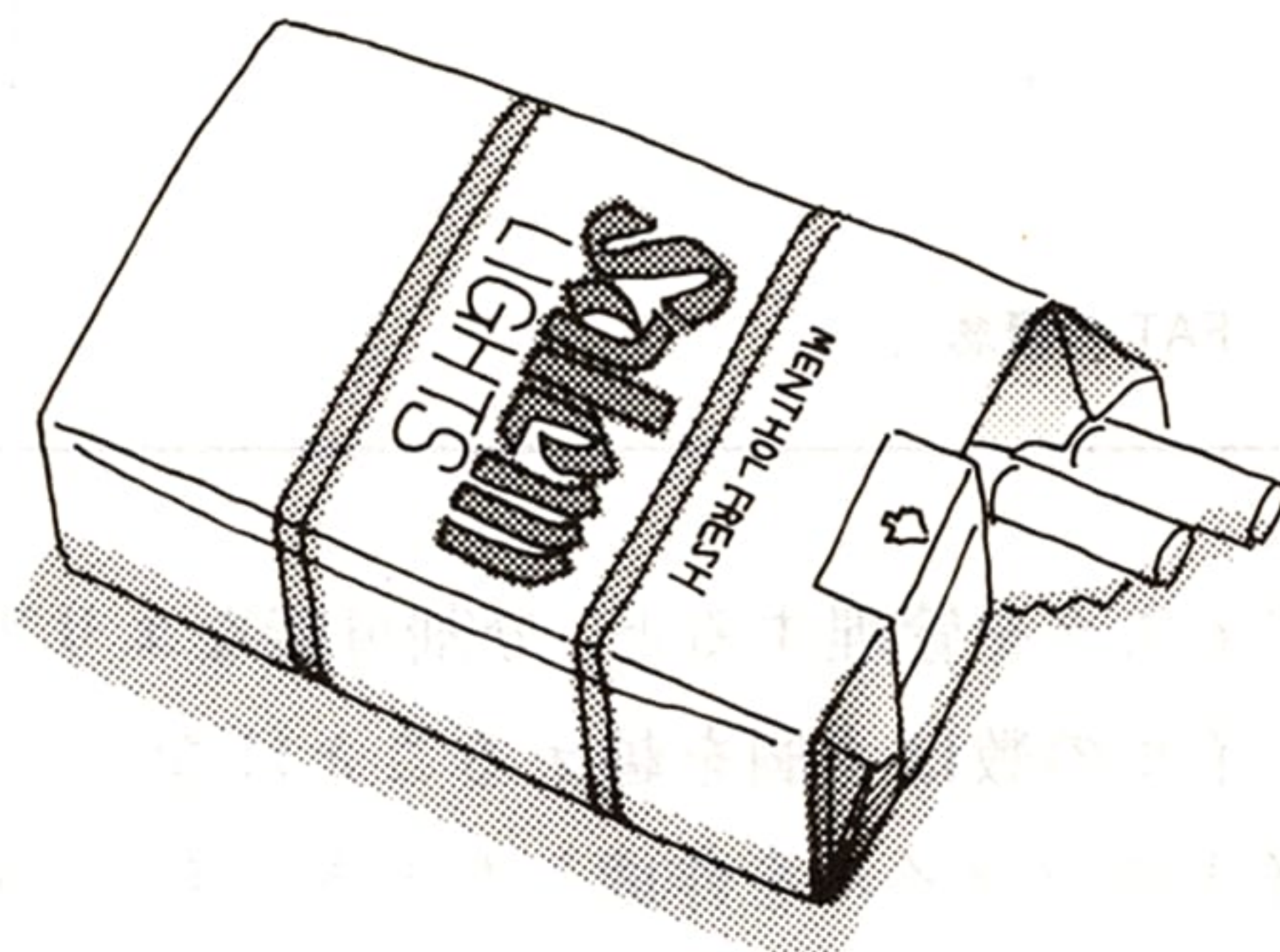
図4.18　FATの概念

このように、2ビットFATで10Mバイトのディスクを管理すると、分割可能なエリア(つまり管理可能なクラスタ)は4つであり、収容可能なファイルの数は4個を越えることはなく、1クラスタは2560Kバイトになります。つまり、たった1バイトのファイルを作成したときでも、2560Kバイトのディスクメモリを消費するわけです。

4ビットFATになると、分割可能なエリアは16個になり、収容可能なファイルの数は最大16個、1クラスタは640Kバイトになります。2ビットFATのときの2560Kバイトと比べて、事態はだいぶよくなりました。

これが12ビットFATになると、分割可能なエリアは4096個*となり、クラスタサイズは2.5Kバイトにすることが可能になります。ただし2.5Kバイトというのは、単純に割算をした結果であり、実際にどのようなサイズにするかは、MS-DOSのシステムディスクを提供するそれぞれのメーカーの考え方により異なります。PC-9800シリーズの標準フォーマットでは、さきの表4.1にもあるように、8Kバイトに設定されています。

12ビットFATでは、分割可能なエリアは4096個*ですが、これが16ビットFATでは65536個*になり、クラスタのサイズをさらに1桁以上小さくすることが可能になります。ただし、クラスタのサイズは、小さければよいというものではなく、小さくすればするほどファイルの管理作業が繁雑になり、読み／書きのスピードは遅くなります。また、FATが占める領域も大きくなります。したがって、クラスタのサイズは、あくまでディスク容量とのバランスの問題なのです。

10Mバイト程度のディスク容量であれば、12ビットFATで十分ですが、40Mバイト以上にもなると、12ビットFATではディスクメモリの使用効率が非常に悪くなり、16ビットFATが必要になります(表4.1のSCSIインターフェイスにおける12ビットFAT／16ビットFATの切り替えに注目)。以上のようなことが12ビットFAT／16ビットFATの概念です。

＊ 実際にクラスタに使われる個数は、これより10個ほど少ない。

4.5 ハードディスクの運用法

ハードディスクは、大容量であることと、ディスク(メディア)の交換ができないことから、フロッピーディスクの場合とは違った運用法が必要になります。その重要なことは、次の2点です。

- **ファイルの収容には必ず階層構造をとること**
- **ファイルのバックアップを必ず行うこと**

本節は、ハードディスクシステムを扱うために必須の、これら2つの基本を中心に実際の運用法について解説しましょう。なお、ハードディスクドライブの電源や、コンピュータシステムの電源を切る際の注意 —— 必ずMS-DOSのコマンドレベルに下(お)りて、STOPキーを押してから電源をOFFにする —— は、本章の4.1で述べていますので、その知識はあるものとします。

■ 必須の階層構造

階層ディレクトリの知識のない人は、ハードディスクを勝手に使っては困ります。もちろん、ただのオペレータであれば話は別ですが、本書の読者の多くは、コンピュータを活用しようとするユーザーであると思われますので、MS-DOSのコマンドなどを使って、ファイルのさまざまな操作を行ったりするでしょう。そのようなユーザーには、階層ディレクトリの知識は必須です。

フロッピーディスクであれば、1枚ごとに、各アプリケーション単位、仕事単位、個人単位などに分けて保管することができます。しかしハードディスクではそうはいきません。そこで階層構造をとることが絶対的に必要になります(階層構造の基礎知識は「入門MS-DOS」を参照)。

前節で述べたように、大容量のハードディスクであれば、複数のパーティションに分割することも(拡張フォーマットの場合)ファイル管理の手段の1つです。しかし、その1つのパーティションも、かなりの大容量であり、そのなかでは、やはり階層構造によるファイル管理を行わなければなりません。

ハードディスクにおける階層構造の基本形

ハードディスクに限らず、すべてのディスクにおいて、ルートディレクトリにはファイルを置かないことが大原則です。どうしてもやむを得ない特別なファイル以外は、ルートディレクトリ上にファイルは存在しない、という状態がまず第一の基本です。

その〝特別なファイル〟とは、システムディスクの場合、「COMMAND.COM」「CONFIG.SYS」「AUTOEXEC.BAT」の3つです。この3つのファイルは、ルートディレクトリ上に置かなければ機能しない、MS-DOSシステムの特別なファイルです。また、各種のデバイスドライバのなかには、ルートディレクトリに置かなければ機能しないものがあるかもしれません。また、ワープロや日本語入力

システム(FEP)の辞書ファイルなども、ルートディレクトリに置かなければならないものもあるでしょう。こういったものは仕方ありませんので、ルートディレクトリに収容することになります。

さて、そのほか一般の諸々のファイルは、サブディレクトリ上に収容することになりますが、その分類整理の仕方は、各ユーザーの使い方によって理想的な形が異なってくるでしょう。次の図は、システムディスクにおける階層構造の基本的な考え方を示したものです。

ルート
ディレクトリ
COMMAND.COM
CONFIG.SYS
AUTOEXEC.BAT
ルートに置かなければ動作しない各種のデバイスドライバや辞書ファイルなど
ルートディレクトリ上に直接存在するファイルはこれらのみ
サブディレクトリ上に置くことが可能なものは、図の下に記した各ディレクトリに収容する
各種のデバイスドライバのディレクトリ
MS-DOS関係のディレクトリ
FEPとワープロ関係のディレクトリ
ビジネスソフト「A」のディレクトリ
ビジネスソフト「B」のディレクトリ
その他各種のディレクトリ
その他各種のディレクトリ
それぞれのユーザーの使い方に合せて、さらにこれらの上にサブディレクトリや各種のファイルを収容する。
上に示したファイルを除き、ルートディレクトリ上には、これらの「ディレクトリ」以外に、直接収容されている「ファイル」がないことに注目！

図 4.19　ハードディスクにおける階層構造の基本的な考え方

図に書かれているのは、全体の階層を構築するための基礎になる部分であり、当然、それぞれのサブディレクトリの先にも、各ユーザーの使い方に合わせた階層構造が作られます。

階層構造は、多くのファイルを組織的・系統的に管理できると同時に、各ディレクトリ間が〝絶縁〟されることにより、ファイルの保管上の安全性が非常に高くなります。ハードディスクを効果的に活用するための基礎は、いかにうまく整理された階層ディレクトリを構築するかにかかっているのです。

■ ハードディスクのバックアップ

ハードディスクは、信頼性の非常に高い装置ですが、やはりフロッピーディスクの場合と同様、重要なファイルはバックアップコピーをとっておくことが必要です。大型コンピュータなどでは、普通、MT 装置(磁気テープによる記憶装置)でバックアップを行いますが、パーソナルコンピュータの場合は、MT 装置が普及していないため、一般的にはフロッピーディスクで行います。

ハードディスク上のファイルのバックアップには、フロッピーディスク間のバックアップとは少し違った部分があります。フロッピーディスク間のバックアップであれば、受け側にそれと同じ容量のフロッピーディスクを使えば、COPY コマンドや DISKCOPY コマンドで、すべてのファイルを必ずバックアップすることができますが(コピー先のディスクがいっぱいになって溢れることはない)、ハードディスクではそうはいきません。たとえ、ある特定のディレクトリ上のファイルだけを対象にしても、1 枚のフロッピーディスクに収容できるとは限りません。一般のフロッピーディスクは、約 1.2M バイトの容量ですから、コピーするファイルの容量の合計がそれ以上になれば、1 枚には収らないのは当然です。

フロッピーディスク間のファイルのコピーは、普通、COPY コマンドを使って、

A＞COPY　*.*　B：⏎　　とか　　A＞COPY　¥WP¥BUN　B：⏎

などを実行すれば、必ず 1 枚のフロッピーディスクに収めることができますが、ハードディスクの場合は、複数のフロッピーディスクにまたがる場合もあります。その場合は少々やっかいなことになります。1 枚目のフロッピーディスクには、上のような COPY コマンドでバックアップを行えば、1 枚のディスクの容量の限界までファイルを詰め込むことができます。問題は 2 枚目以降のコピーです。2 枚目には同じコマンド —— すべてのファイルを一括コピーする COPY コマンド —— は使えません。COPY コマンドには、どのファイルがコピー済みで、次はどのファイルからコピーするのかなどの、コピー「済み／未」を管理する機能がないからです。1 枚目にコピーされたファイルを調べて、それ以外のファイルを 1 つ 1 つ名指しでコピーする以外にありません。つまり、通常の COPY コマンドを使って、複数のフロッピーディスクにまたがるようなバックアップコピーを、日常的な作業として行うことは現実的ではないのです。

バックアップおよび復元コマンド

そこで、大容量のハードディスクのバックアップに対応するために用意されているコマンドが、「BACKUP コマンド」と「RESTORE コマンド」です。この2つのコマンドは互いに組になっているコマンドです。BACKUP コマンドによりバックアップコピーを行っていれば、もしコピー元のファイルに事故があった場合は、RESTORE コマンドにより事故ファイルの復元を行うことができるようになっています。

BACKUP コマンドは、COPY コマンドの機能に、条件コピーや選択コピーを始めとするいろいろな機能を付加し、さらに階層ディレクトリのコピーをサポートしたものと考えてよいでしょう。ただし、BACKUP コマンドでコピーしたものは、RESTORE コマンドでなければ復元できません。これは、BACKUP コマンドによるバックアップが、特殊なフォーマットでコピーされるためです。

BACKUP コマンドでとくに重要なのは、実際にバックアップコピーを行ったコピー元のファイルに、「コピー済み」のマークを付ける機能です。これは、6章の「ATTRIB コマンド」の実行例(298ページの図 6.20a)のリードオンリー・アトリビュートを示す「R」のすぐ右にある「A」が、その「コピー済みマーク」である「アーカイブ・アトリビュート」(**アーカイブ属性**)なのです。

```
A>ATTRIB  COMMAND.COM ⏎
    R A     A:¥COMMAND.COM
      ↑
```

アーカイブ属性であることを示すマーク

アーカイブ属性(Archive：記録とか書類とかの保管所というような意味)が付いているファイルは、バックアップコピーがまだ行われていないことを示しています。つまり、〝書庫に未保管〟のマークがついていると考えればよいでしょう。BACKUP コマンドは、このアーカイブのアトリビュート(属性)を操作する(コピーしたコピー元ファイルのアーカイブ属性を取り去る)ことにより、バックアップ済みのファイルと、そうでないファイルとの区別をしています。そのアーカイブ属性の変化の様子を次ページの**図 4.20** に示しましょう。

このアトリビュートを操作しながらバックアップコピーを行い、1枚目のコピー先フロッピーディスクがいっぱいになれば、2枚目のディスクをセットするようにメッセージを表示し、そのディスクがセットされると、次のアーカイブ属性の付いているファイルからバックアップコピーを続けます。3枚目以降のディスクも同様です。

また、バックアップコピーのディスクから、事故に遭ったコピー元のファイルを自動的に復元する場合にも、このアーカイブ属性が利用されます(くわしくは後述)。ATTRIB コマンドについては、6章の「ディスクファイルに関する操作」の項(297ページ)を参照してください。

さて、BACKUP／RESTORE の各コマンドが、ファイルのアーカイブ属性を利用して、保存／復元を行うことがわかりましたので、さっそく実行例を示しましょう。

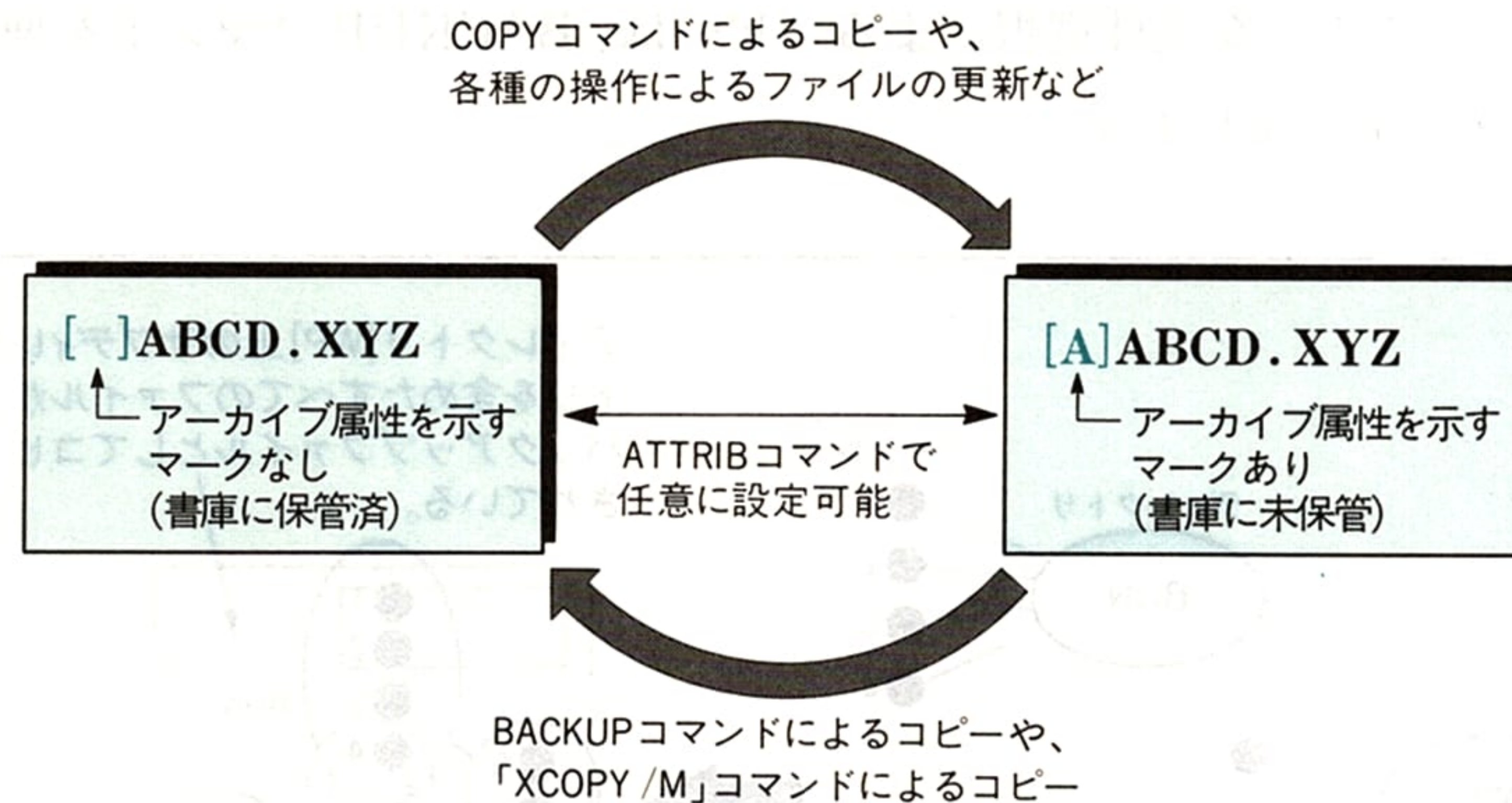



図 4.20　アーカイブ属性の各種コマンドによる変化

BACKUP コマンドを使ったバックアップ

BACKUP(保存)コマンド、RESTORE(復元)コマンドには、条件コピーや選択コピーなど、スイッチによるいろいろな機能がありますが、ここでは、〝日課〟として行うための、もっとも一般的なバックアップの方法を紹介します。さらに万が一、もとのファイルが事故に遭った場合(誤って削除されたり、書き換えられたような場合)を想定して、それらのファイルを復元する方法についても紹介しましょう。

ここで実習するマシンのディスクシステムは、ドライブA：B：がハードディスク、ドライブC：D：がフロッピーディスクであり、ドライブA：上のファイルをドライブC：上のフロッピーディスクにバックアップコピーするという想定です。

さて、ドライブA：のハードディスクには、6章の270ページの図6.1の階層ディレクトリと同じ状態で、各種のファイルが収容されているとしましょう。ここでは、そのなかの[¥WP]上の、すべてのサブディレクトリを含めた、すべてのファイルをバックアップする方法を実習します。ディレクトリ[WP]は、ワープロの文書ファイルを収容するディレクトリであり、次ページの図 **4.21** に示すように、サブディレクトリ[BUN](新松用)、および[JXW](一太郎用)が収容されています。

このハードディスク上で、毎日、いろいろな文書の作成作業が行われます。これら2つの文書ファイル用のディレクトリには、新しく作成された文書ファイルが収容されたり、既存の文書ファイルが書き換えられ(更新され)たり、さまざまな操作が行われます。そこで、1日の作業が終わった時点で、その日に新たに作成されたり、何らかの変化があったファイルを、〝日課〟としてバックアップすることが必要です。

つまり、ディスク上やディレクトリ上のすべてのファイルを毎回バックアップコピーするのではな

く、そのなかの新規に作成されたファイルや、更新が行われたファイルのみをバックアップコピーの対象にするわけです。このような条件選択によるコピーは、BACKUP コマンドを使えば可能です。そのときの動作を次の図で示しましょう。

ファイル
ディレクトリ
BUN
1
2
3
4
ディレクトリ
WP
5
6
ディレクトリ
JXW
7
8
9
(ワープロ作業が行われる)
ディレクトリ[WP]上のサブディレクトリを含めたすべてのファイルが、バックアップファイルとしてコピーされている。
BUN
WP
JXW
〈バックアップ用フロッピーディスク〉
(1日の作業が終わり、文書ファイル「1」「2」が更新され、新規ファイル「10」「11」「12」が作成されたとする)
BACKUPコマンドを実行する
BUN
WP
JXW
10
11
12
1日の作業が終わった時点で、ハードディスク上のファイルのバックアップを行う。
前回のバックアップ以降に更新されたり、新規作成されたファイルのみがバックアップされる

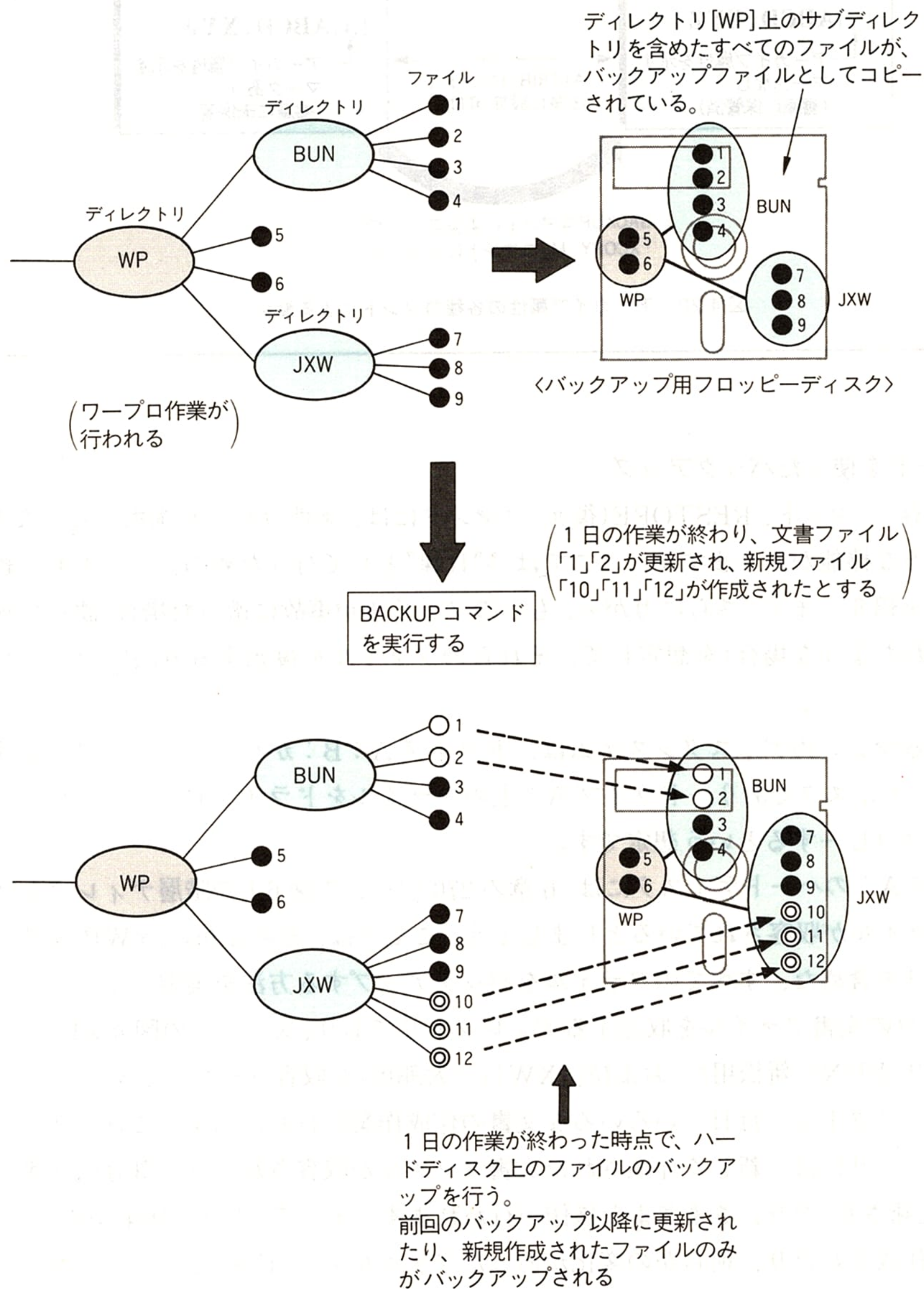



図 4.21　変化のあったファイルのみがバックアップされる仕組み

本日の作業を始める前の時点のハードディスク上のディレクトリ[WP]は、図 4.21 の上側のような状態であったとしましょう。このディレクトリ[WP]上のファイルは、サブディレクトリ上のものを含めて、そのすべてが昨日のバックアップ作業で、フロッピーディスクに収容(バックアップコピー)されています。

そして本日の作業では、[BUN]上の①②の 2 つのファイルが更新され、[JXW]上の⑩⑪⑫の 3 つのファイルが新規に作成されました(一太郎バージョン 3 では、1 つの文書ファイルは 3 つのファイルからなる)。作業終了後のバックアップ作業は、自動的に図 4.21 の下側のように行われるはずです。

実行例に使う実際のハードディスクには、ディレクトリ[BUN]上にも[JXW]上にも次の図 4.22 に示すような多くのファイルがあり、それらは昨日までのバックアップ作業で、2 枚のフロッピーディスクにバックアップされています。

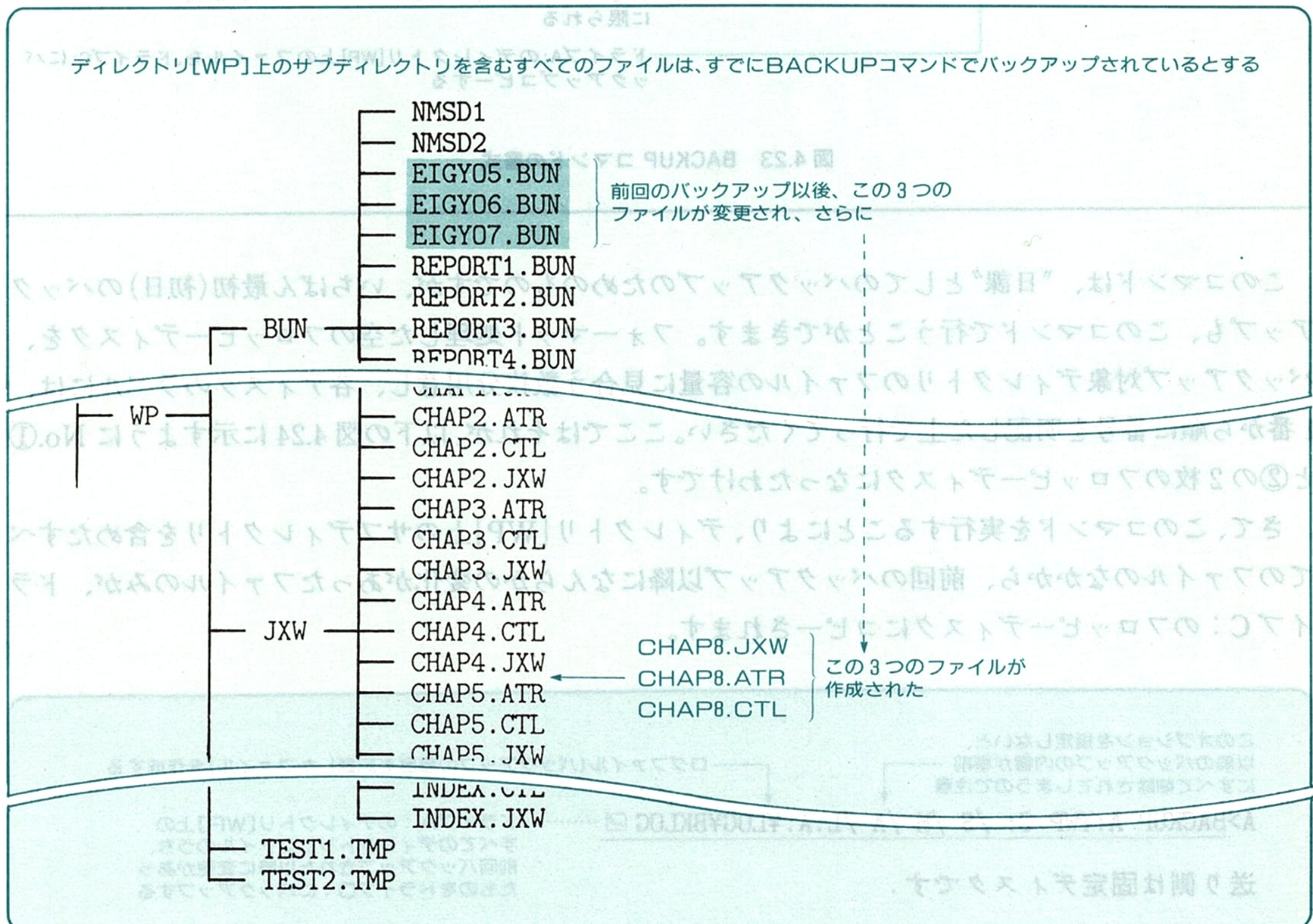



図 4.22　バックアップの対象となる実際のディレクトリの内容

では、「変化のあったファイルのみ」を図 4.21 のように自動的にバックアップコピーするための BACKUP コマンドの書式と、その実行例を示しましょう。

A>BACKUP A:¥WP C: /S /M /A /L:A:¥LOG¥BKLOG

- A:¥LOG¥BKLOG — ドライブA:上のディレクトリ[¥LOG]に、「BKLOG」というファイル名で履歴を記録しておく
- /L — バックアップコピーを行った履歴を記録しておくためのスイッチ
- /A — バックアップ先のディスクの内容を事前に削除せずに、バックアップファイルをコピーするためのスイッチ。このスイッチを付けない場合、バックアップ先のディスクの内容は、バックアップの前にすべて削除される
- /M — 前回のバックアップ以降に変化のあったファイル（つまりアーカイブ属性の付いているファイル）のみを、バックアップコピーするためのスイッチ。このスイッチを付けない場合、指定ディレクトリ上のすべてのファイルをバックアップコピーする
- /S — 指定ディレクトリ上のサブディレクトリを含めたすべてのファイルをバックアップの対象にするためのスイッチ。このスイッチを付けない場合、指定ディレクトリ上の直接のファイルのみに限られる
- A:¥WP C: — ドライブA:のディレクトリ[WP]上のファイルを、ドライブC:にバックアップコピーする

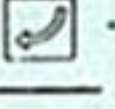



図 4.23 BACKUP コマンドの書式

このコマンドは、〝日課〟としてのバックアップのためのものですが、いちばん最初(初日)のバックアップも、このコマンドで行うことができます。フォーマット処理した空のフロッピーディスクを、バックアップ対象ディレクトリのファイルの容量に見合う数枚分用意し、各ディスクのラベルには、1番から順に番号を明記した上で行ってください。ここではそれが、以下の図 4.24 に示すように No.①と②の2枚のフロッピーディスクになったわけです。

さて、このコマンドを実行することにより、ディレクトリ[WP]上のサブディレクトリを含めたすべてのファイルのなかから、前回のバックアップ以降になんらかの変化があったファイルのみが、ドライブC：のフロッピーディスクにコピーされます。


```
A>BACKUP A:¥WP C: /S /M /A /L:A:¥LOG¥BKLOG

送り側は固定ディスクです.

最後のバックアップディスクをドライブ C: に挿入してください.
準備ができたらどれかキーを押してください.

*** ファイルをドライブ C: にバックアップします. ***
ディスク番号: 02
```

```
¥WP¥BUN¥EIGYO5.BUN
¥WP¥BUN¥EIGYO6.BUN
¥WP¥BUN¥EIGYO7.BUN
¥WP¥JXW¥CHAP8.ATR
¥WP¥JXW¥CHAP8.CTL
¥WP¥JXW¥CHAP8.JXW

A>TYPE ¥LOG¥BKLOG

[1989-9-15 23:03:36]
 1  /WP/TEST1.TMP
 1  /WP/TEST2.TMP
 1  /WP/BUN/EIGYO5.BUN
 1  /WP/BUN/EIGYO6.BUN
 1  /WP/BUN/EIGYO7.BUN
 1  /WP/BUN/FIX.BUN
 1  /WP/BUN/NMSD1
 1  /WP/BUN/NMSD2
 2  /WP/JXW/INDEX.ATR
 2  /WP/JXW/INDEX.CTL
 2  /WP/JXW/INDEX.JXW

[1989-9-16 18:09:50]
 2  /WP/BUN/EIGYO5.BUN
 2  /WP/BUN/EIGYO6.BUN
 2  /WP/BUN/EIGYO7.BUN
 2  /WP/JXW/CHAP8.ATR
 2  /WP/JXW/CHAP8.CTL
 2  /WP/JXW/CHAP8.JXW

A>
```


図 4.24　BACKUP コマンドによるバックアップコピーの実行例

このように、今回のバックアップは、2 枚目のフロッピーディスクにコピーされました。このような、変化のあったファイルの選択コピーは、さきほど述べたアーカイブ属性を利用して行われます。バックアップコピーが行われた時点で、コピー元のファイルのアーカイブ属性は解除されますので、次のバックアップでは、その後に何らかの変化があり、新たにアーカイブ属性が付けられたファイルのみがバックアップの対象になります。

実行例の「/L：」スイッチにより、バックアップファイルに関する履歴が残されますので、図 4.24 のなかで、そのファイルをタイプアウトしています。この履歴ファイルは、バックアップ先のファイルの状況を確認するための大切な資料です。

このようなバックアップ作業を、〝日課〟として行うことにより、ハードディスクの任意のディレクトリ上の最新の内容を、常にバックアップしておくことができます（BACKUP コマンドには、そのほかのスイッチによるいくつかの条件選択の機能もあるが実行例は省略する）。

なお参考までに、BACKUP コマンドのそのほかのスイッチとその機能を示しておきます。

/D：　………ファイルの作成日付が指定した日付以降であるファイルのみをバックアップする
/T：　………ファイルの作成時刻が指定した時刻以降であるファイルのみをバックアップする
/P　…………バックアップされたファイルの数が、バックアップ先のルートディレクトリ（つねにルートディレクトリにコピーされる）の収容可能ファイル数を超えると（ディスクの容量がいっぱいになる前に）、自動的にサブディレクトリを作成し、そこにバックアップコピーを行う

例：　A＞BACKUP　A：¥WP　C：　/S /M /A /P /D：89-09-15 /T：13：00⏎

いずれにしても、BACKUP コマンドでコピーされたファイルは、次項の RESTORE コマンドで復元しなければ使用することはできないことに注意してください。

RESTORE コマンドによる事故ファイルの復元

では、もとのハードディスクの、ディレクトリ[¥WP¥JXW]上のいくつかのファイルを誤って削除したり、内容を誤って書き換えてしまった場合に、もとの状態に復元する作業を示しましょう。

前項の実行例の日に作成した、ハードディスク上のディレクトリ[JXW]上の3つのファイル⑩⑪⑫（一太郎バージョン3では3つで1つの文書ファイル）を誤って削除してしまったとしましょう。作業を行った日には、〝日課〟として必ずバックアップを行っていますので、そのフロッピーディスクから、RESTORE コマンドを使って自動的に復元することが可能です。そのときの動作を次ページの図4.25で示します。

この図のように、ハードディスク上で、前回のバックアップ以降に変化のあったファイル ―― つまり、削除されたファイル、およびアーカイブ属性が付いたファイル ―― だけを、バックアップディスクから自動的に選択して復元することができます。

このような復元作業を実行するための RESTORE コマンドの書式（図4.26）とその実行例（図4.27a）を201ページと202ページに示しましょう。

ここで注意しなければならないことは、「/M」スイッチを付けることにより、前回のバックアップ以降に変化のあったファイルのみが、自動的に復元されますので、もし前回のバックアップ以降に、別のファイルの更新などを行っていれば、それらは、バックアップディスク上のファイル（つまり前回までの旧ファイル）に〝復元〟される（逆もどりする）ことになります。そこで、誤って削除したファイルの復元作業であれば、更新されたファイルが旧ファイルにもどらないように、その作業の前に前項で行ったバックアップ処理を行っておくことが必要です。

では、この RESTORE コマンドを実行してみましょう（202ページの図4.27a）。

JXW
7
8
9
10
11
12
誤って、この3つの
文書ファイルを消してしまった

RESTOREコマンド
を実行する

WP
BUN
1
2
3
4
5
6
JXW
7
8
9
10
11
12

〈バックアップ用フロッピーディスク〉
BUN
WP
JXW

前回のバックアップ以降に、なんらかの
変化があったハードディスク上のファイ
ルに対してのみ、復元作業が行われる

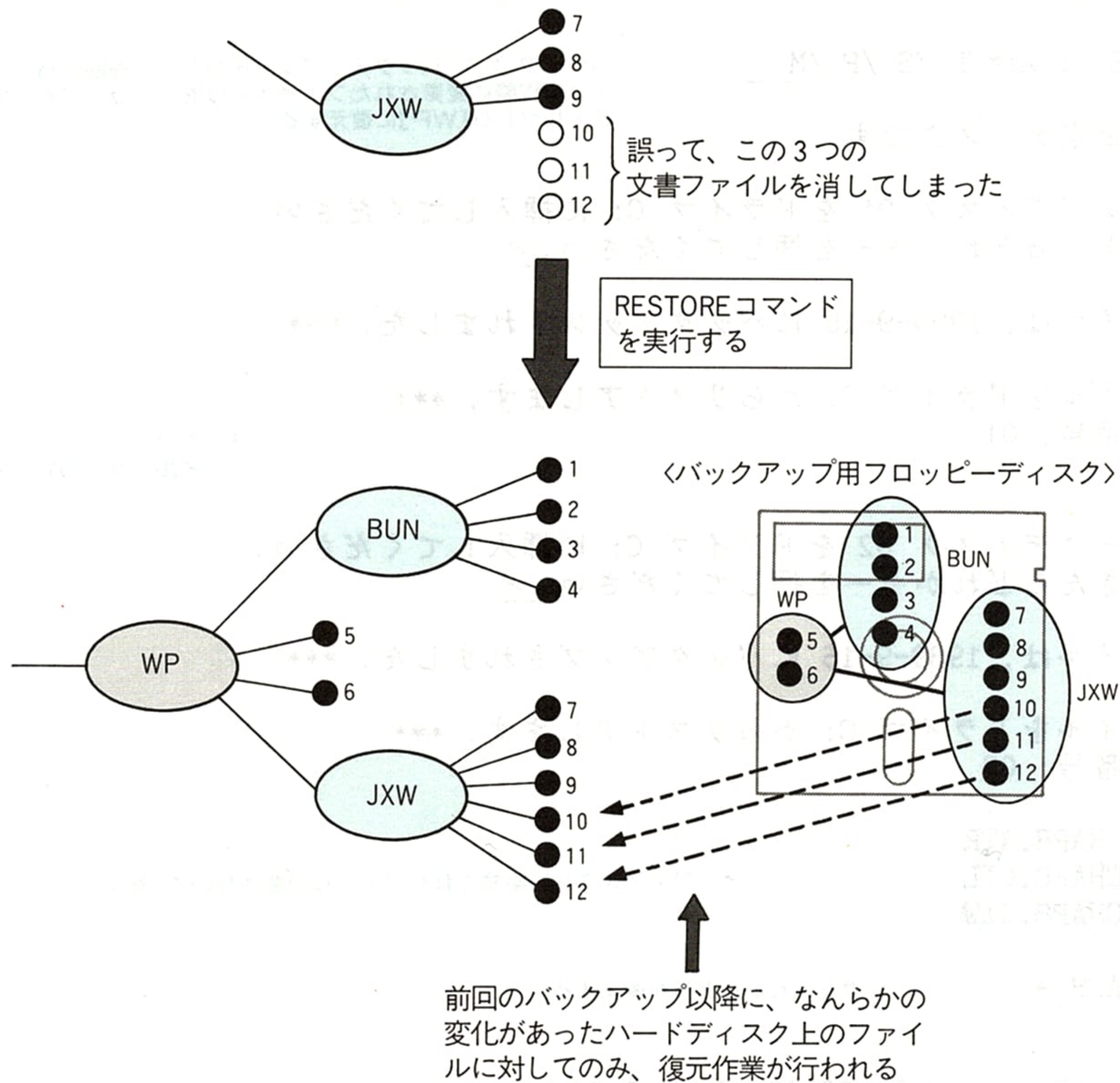



図4.25　誤って削除したファイルが復元される仕組み

```
A>RESTORE C: A:¥WP /S /P /M
```

- /M：前回のバックアップ以降に変化のあったファイルのみを復元するためのスイッチ。このスイッチを付けない場合、バックアップディスク上のすべてのファイルの、それぞれの最新世代のものが復元される
- /P：ファイルのアトリビュートが「隠しファイル」あるいは「リードオンリーファイル」の場合は、復元する／しないの問い合わせをさせるためのスイッチ
- /S：指定ディレクトリ上のサブディレクトリを含めたすべてのファイルを復元の対象にするためのスイッチ。このスイッチを付けない場合、指定ディレクトリ上のファイルのみに限られる
- C:　A:¥WP：ドライブC:のバックアップディスクから、ドライブA:上の[¥WP]上のファイルを復元する

図4.26　RESTOREコマンドの書式

```
A>DEL CHAP8.*⏎ ……ディレクトリ[JXW]のファイルを誤って消してしまった

A>RESTORE C: A:¥WP /S /P /M⏎ ……ドライブC：のバックアップファイルから、前回のバックアップ以降に変更されたファイルだけを、ドライブA：のディレクトリ[WP]に復元する

受け側は固定ディスクです.

バックアップディスク 01 をドライブ C: に挿入してください.
準備ができたらどれかキーを押してください.⏎

*** ファイルは, 1989-9-15 にバックアップされました. ***

*** ファイルをドライブ C: からリストアします. ***
ディスク番号: 01
                                        バックアップディスク①には復元するファイルがないので何もしない

バックアップディスク 02 をドライブ C: に挿入してください.
準備ができたらどれかキーを押してください.⏎

*** ファイルは, 1989-9-16 にバックアップされました. ***

*** ファイルをドライブ C: からリストアします. ***
ディスク番号: 02

¥WP¥JXW¥CHAP8.ATR
¥WP¥JXW¥CHAP8.CTL ……バックアップディスク②に削除されたファイルがあったので、復元する
¥WP¥JXW¥CHAP8.JXW

A>DIR CHAP8.*⏎ ……復元されたかどうか確認する
        ⋮
CHAP8    ATR      830  89-09-16  13:24
CHAP8    CTL     1796  89-09-16  13:24   削除されたファイルが復元された
CHAP8    JXW    41135  89-09-16  13:24
        ⋮
A>
                                        ＊DIRコマンドのメッセージは省略してある
```

図 4.27a　RESTORE コマンドによる事故ファイル復元の実行例①

このように、バックアップディスクが複数枚の場合は、目的のファイルが、何番目のディスクにバックアップされているかが判別できないため、１番目のディスクから順に処理するようにメッセージが表示されます。メッセージにしたがって全部のディスクを処理すれば、最終的には、事故ファイルが最新世代のバックアップファイルで無事に復元されます(実際にはログファイルを調べることにより、目的のファイルがバックアップされているディスクの番号がわかるので、そのディスクをセットすれば、警告のメッセージが表示されるものの復元は正しく行われる)。

ここでもう一例、事故ファイルの復元を行ってみましょう。ディレクトリ[BUN]上のファイルから、ファイル名に「EIGYO」という文字列を含むすべてのファイルを誤って削除した場合の復元です。

```
A>DEL EIGYO*.*⏎ ……………ディレクトリ[BUN]のファイルを誤って消してしまった

A>RESTORE C: A:¥WP /S /P /M⏎ ……………図4.27aと同様に復元する
受け側は固定ディスクです.

バックアップディスク 01 をドライブ C: に挿入してください.
準備ができたらどれかキーを押してください.⏎

*** ファイルは, 1989-9-15 にバックアップされました. ***

*** ファイルをドライブ C: からリストアします. ***
ディスク番号: 01

¥WP¥BUN¥EIGYO5.BUN
¥WP¥BUN¥EIGYO6.BUN ……………バックアップディスク①の古いファイルが復元されてしまう
¥WP¥BUN¥EIGYO7.BUN

バックアップディスク 02 をドライブ C: に挿入してください.
準備ができたらどれかキーを押してください.⏎

*** ファイルは, 1989-9-16 にバックアップされました. ***

*** ファイルをドライブ C: からリストアします. ***
ディスク番号: 02

警告! ファイル ¥WP¥BUN¥EIGYO5.BUN はバックアップ後に変更されています.
ファイルを入れ替えますか <Y/N>? Y⏎
¥WP¥BUN¥EIGYO5.BUN

警告! ファイル ¥WP¥BUN¥EIGYO6.BUN はバックアップ後に変更されています.
ファイルを入れ替えますか <Y/N>? Y⏎
¥WP¥BUN¥EIGYO6.BUN

警告! ファイル ¥WP¥BUN¥EIGYO7.BUN はバックアップ後に変更されています.
ファイルを入れ替えますか <Y/N>? Y⏎
¥WP¥BUN¥EIGYO7.BUN
                         バックアップディスク②のファイルが
                         最新なので、こちらを復元する

A>DIR EIGYO*.*⏎ ……………復元されたかどうか確認する
        :

EIGYO5   BUN    13525  89-09-16   16:34
EIGYO6   BUN    18614  89-09-16   17:30  最新ファイルが復元された
EIGYO7   BUN     8789  89-09-16   18:02
        :
A>
                         ＊DIRのメッセージは省略してある
```

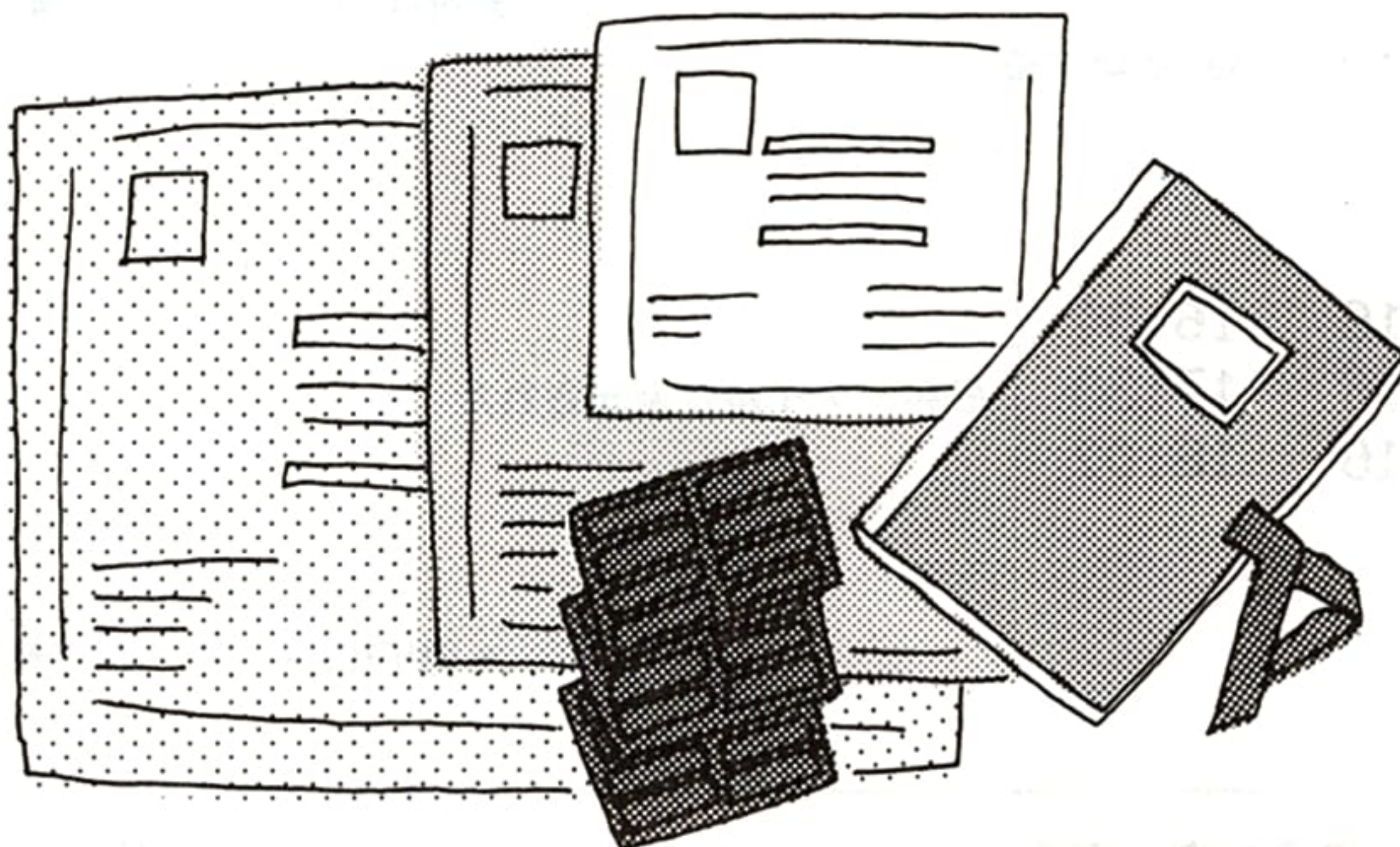
図 4.27b　RESTORE コマンドによる事故ファイル復元の実行例②

さきほどと同様に、1番目のバックアップディスクから順に処理していくことにより、自動的に必要なファイルが選択され、復元処理が行われます。ただし、同じファイル名でバックアップされている旧世代のファイルが存在している場合には、それらから順に復元されることになります。しかし、それらは新しい世代のファイルに順次復元し直されていき、すべてのディスクを処理した時点では、結局、最新世代のファイルが復元されています。

ここで示したBACKUP／RESTOREの各コマンドの使い方は、日常もっとも一般的に行われる形式を想定した一例ですが、これらのコマンドは、各種の条件選択のスイッチにより、さまざまな使い方が可能です。ここではそれらの具体的な解説は行いませんが、条件選択の機能の基本は、各ファイルのアーカイブ属性、あるいはファイル作成日時のスタンプ機能を利用しています。日時のスタンプは、強制的に変更することはできませんが、アーカイブ属性は、ATTRIBコマンドを使って任意に変更することが可能です。したがって、BACKUP／RESTOREコマンドを実行する際に、必要に応じて、あらかじめATTRIBコマンドでアーカイブ属性を意識的に操作しておくことにより、さらに細かい意図的なバックアップや復元の操作を行うことが可能です。

なお参考までに、RESTOREコマンドのそのほかのスイッチとその機能を示しておきます。

/A：　………ファイルの作成日付が、指定した日付以降であるファイルのみを復元する
/B：　………ファイルの作成日付が、指定した日付以前であるファイルのみを復元する
/L：　………ファイルの作成時刻が、指定した時刻以降であるファイルのみを復元する
/E：　………ファイルの作成時刻が、指定した時刻以前であるファイルのみを復元する
/N　…………復元先に存在しないファイルのみを復元する

例：　A＞RESTORE　C：　A：¥WP　/S /P /M /A：89-09-15　/L：13：00↵

サブディレクトリを含めたファイルのコピー　XCOPY コマンド

MS-DOS のバージョン 3.*x* 以降のシステムディスクには、階層ディレクトリに対応したファイルのコピーが行える「XCOPY コマンド」(外部コマンド)が付属しています。従来の COPY コマンドでは、1 回の実行で、指定した任意のディレクトリ上に直接存在するファイルしかコピーできませんでしたが、XCOPY コマンドでは、指定したディレクトリに連なるすべての階層構造と、それらに存在するすべてのファイルをそっくりコピーすることができます。その動作を次の図で示しましょう。

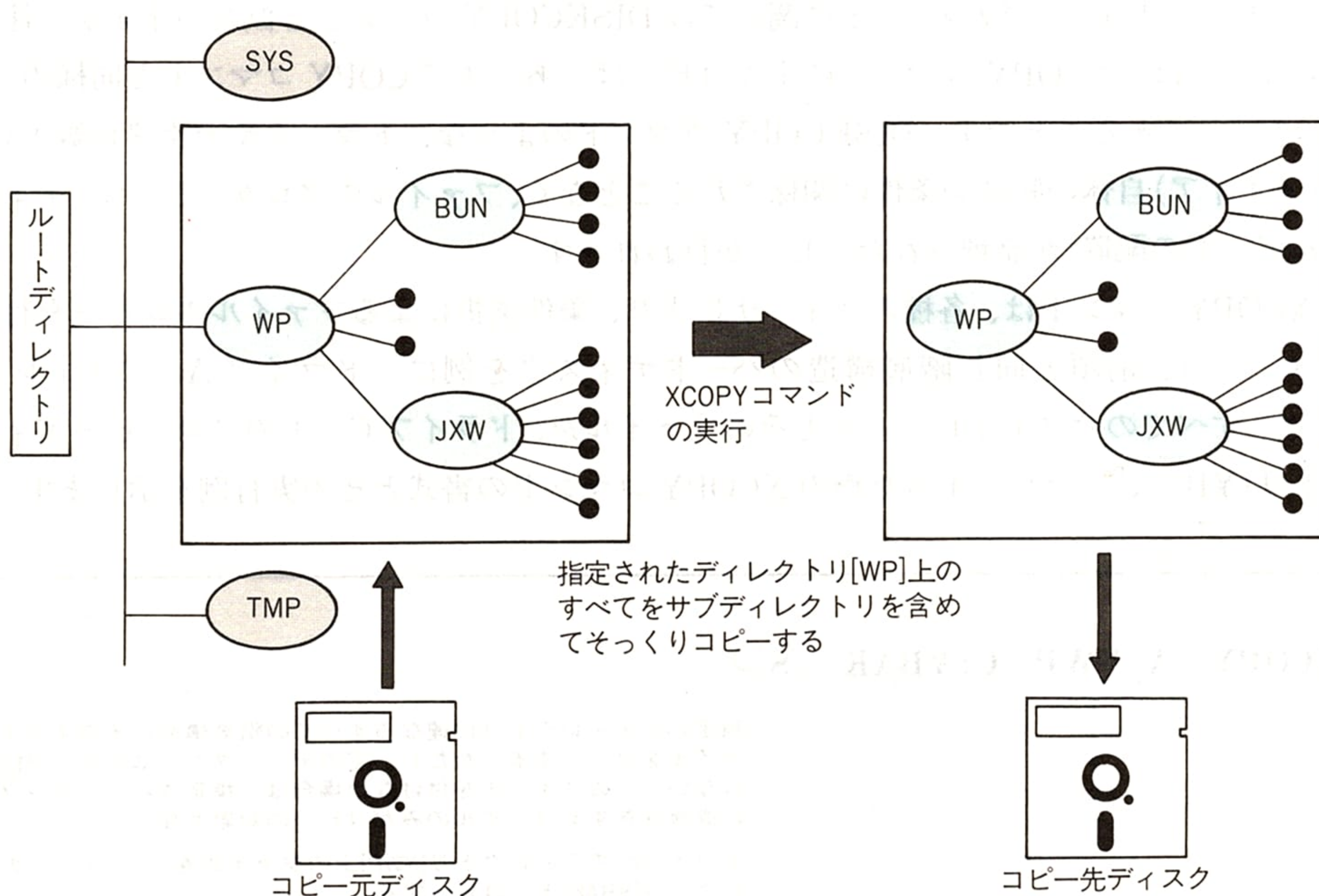



図 4.28　XCOPY コマンドによるディレクトリのコピー

これは、ルートディレクトリ上のサブディレクトリ[WP]を指定して XCOPY コマンドを実行する例ですが、この図のように、ディレクトリ[WP]に連なるすべてのサブディレクトリとファイルを、1 回の実行でコピーできることを表しています。コピー先のディスク上のディレクトリ[BUN]や[JXW]は自動的に作成され、その上にそれぞれのファイルがコピーされます。ただし、指定したディレクトリ[WP]は、XCOPY コマンドを実行する際のコマンドラインに、コピー先のディレクトリ名を指定しておくことにより、その指定した名前で作成されます。その指定がなければ、自動的には作成されません。

参考：このときに、指定したディレクトリを、コピー先のディスク上に「作成するか／しないか」の問い合わせがあるが、「XCOPY.EXE」を MCOPY.EXE にリネームしたもので実行すれば、コピー元がディレクトリであるか、ファイルであるかを自動的に判別してくれるので問い合わせは発生しない。この XCOPY コマンドは作りが少々変則的なコマンドであり、結局、『XCOPY.EXE は MCOPY.EXE にリネームして、MCOPY コマンドとして使った方が便利である』。

さて本論にもどり、もしコピー元のディレクトリに、ルートディレクトリ[¥]を指定した場合には、そのディスク上の階層構造のすべて、ファイルのすべてが、そっくりコピーされることになります。この場合、「そっくりコピーされる」ことに関しては DISKCOPY コマンドと似ていますが、注目しなければならない点は、XCOPY コマンドによるコピーは、あくまで COPY コマンドと同様の「ファイル単位のコピー」であることです。DISKCOPY コマンドのような、トラックやセクタの数といった、ディスク（メディア）自体の物理的条件に関係されることなく、ファイルのアロケーション（ディスク上のファイルデータの配置）が整理されたコピーが行われます。

さらに XCOPY コマンドは、各種のスイッチにより、条件選択によるファイルのコピーを行うことが可能です。では、前項と同じ階層構造のハードディスクを例に、ドライブ A：のディレクトリ[¥WP]上の、すべてのサブディレクトリとそのファイルを、ドライブ C：上のフロッピーディスクのディレクトリ[¥BAK]にコピーするための XCOPY コマンドの書式とその実行例を示します。

```
A>XCOPY  A:¥WP  C:¥BAK  /S↵
```

- /S：指定したディレクトリに連なるすべての階層構造とそのすべてのファイルをコピーする。ただし、空のディレクトリはコピー（作成）されない。このスイッチを付けない場合は、指定されたディレクトリに直接存在するファイルのみがコピーの対象となる
- A:¥WP C:¥BAK：ドライブA:のディレクトリ[¥WP]上のファイルを、ドライブC:のディレクトリ[¥BAK]上にコピーする

図 4.29　XCOPY コマンドの書式

```
A>XCOPY A:¥WP C:¥BAK /S↵
送り側のファイルを読込み中です．．
A:¥WP¥TEST1.TMP
A:¥WP¥TEST2.TMP
A:¥WP¥BUN¥EIGYO5.BUN
A:¥WP¥BUN¥EIGYO6.BUN
      :
```

A>XCOPY A:¥WP C:¥BAK /S：ドライブA：のディレクトリ[¥WP]上のすべてのファイルをサブディレクトリも含めて、ドライブC：のディレクトリ[¥BAK]上にコピーする

（途中省略）

```
          :
A>DIR C:¥BAK ⏎ ……………コピーされたかどうかディレクトリ[BAK]を確認する
          :
BUN          <DIR>       89-09-17    18:52
JXW          <DIR>       89-09-17    18:52
          :
A>DIR C:¥BAK¥BUN ⏎ ……………コピーされたかどうかディレクトリ[BUN]を確認する
          :
EIGY05   BUN    13525    89-09-16    16:34
EIGY06   BUN    18614    89-09-16    17:30
          :
A>DIR C:¥BAK¥JXW ⏎ ……………コピーされたかどうかディレクトリ[JXW]を確認する
          :
CHAP1    ATR     2310    89-08-20    19:20
CHAP1    CTL     1796    89-08-20    19:20
          :
A>                                   ＊DIRコマンドのメッセージは省略してある
```

図 4.30　XCOPY コマンドによるディレクトリの一括コピーの例

このようにスイッチ「/S」により、指定したディレクトリ上の階層構造とそのすべてのファイルが、ドライブ C：のフロッピーディスクのディレクトリ[BAK]上に、そっくりコピーされています。もしコピー先のディスクに、ディレクトリ[BAK]がない場合には、さきに述べた問い合わせがあり、それに答えることにより、自動的にそのディレクトリが作成されます(MCOPY コマンドにすれば問い合わせはない)。

また、この実行例のスイッチでは、ファイルやサブディレクトリを収容していないサブディレクトリは無視され、そのコピー(作成)は行われません。

参考までに、XCOPY コマンドのそのほかの主要スイッチとその機能を示しておきます。

/A …………アーカイブ属性の付いたファイルのみをコピーする
/M …………/A の機能を持ち、かつ、コピーしたファイルのアーカイブ属性を解除する
/D： ………ファイルの作成日付が、指定した日付以降であるファイルのみをコピーする
/E/S ………サブディレクトリが空であっても、そのサブディレクトリをコピー(作成)する
/P …………1つ1つのファイルに関して、コピーする／しないの問い合わせを行う

例：　A＞XCOPY　A：¥WP　C：¥BAK　/A /D：89-09-15　/S ⏎

■ デバイスドライバの入れ替え

MS-DOSのバージョン3.*x*以降のシステムディスクには､｢キャラクタ系デバイスドライバ｣を､MS-DOSを再起動することなく入れ替えるための「ADDDRVコマンド」(アッド・デバイスドライバ)、「DELDRVコマンド」(デリート・デバイスドライバ)が付属しています(いずれも外部コマンド)。

「キャラクタ系デバイスドライバ」とは、たとえば、日本語入力システムやマウスなどの、データを1バイト単位でやり取りするデバイスドライバのことをいい、それに対して「ブロック系デバイスドライバ｣とは、たとえば、RAMディスクやディスクキャッシュ、EMSメモリなどの、データをブロック単位でやり取りするデバイスドライバのことをいいます。

ADDDRVコマンドは、キャラクタ系デバイスドライバ(以降は、ただデバイスドライバと記す)をMS-DOSシステムに組み込むコマンドであり､DELDRVコマンドは､｢ADDDRVコマンドによって組み込んだデバイスドライバ」を削除するコマンドです。ただし、デバイスドライバの入れ替えに関しては次の制約があります(コマンドの実行に関する制約は後述)。

- 組み込み／削除が可能なものは、「キャラクタ系デバイスドライバ」である
- 別のデバイスドライバを組み込む際には、ADDDRVコマンドによって組み込まれたデバイスドライバを、DELDRVコマンドによって事前にすべて削除しなければならない
- MS-DOSの起動時に組み込まれたデバイスドライバは削除できない

ADDDRV／DELDRVコマンドによるデバイスドライバの入れ替えの概念を次ページの図4.31で示しましょう。

ADDDRVコマンドとDELDRVコマンドの書式を示します。

```
A>ADDDRV　ファイル名↵
```

「ファイル名」は、CONFIG.SYSファイルの「DEVICE指定」の行と同じ書式の、ユーザーが任意に作成するテキストファイルです。このファイルは、任意のドライブの任意のディレクトリに置くことができます。

```
A>DELDRV↵
```

DELDRVコマンドにはパラメータやスイッチはありません。ADDDRVコマンドで組み込んだすべてのデバイスドライバが削除されます。

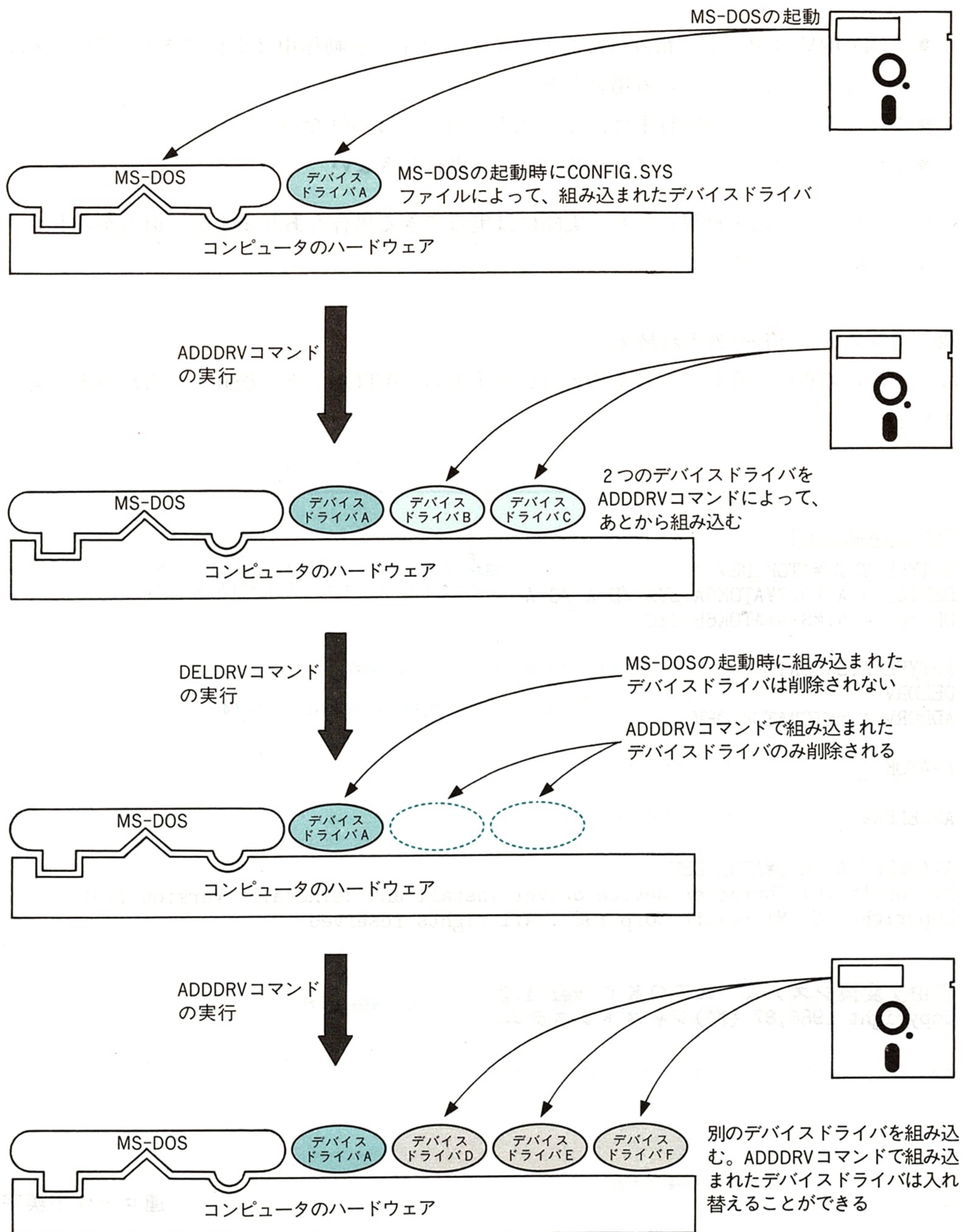



図4.31　ADDDRV／DELDRVコマンドによるデバイスドライバの入れ替えの概念

これら ADDDRV／DELDRV コマンドを実行する際には、次の制約があります。

- ADDDRV コマンドで組み込んだデバイスドライバが動作中は実行できない（たとえば日本語入力モードになっている場合など）
- これらのコマンドの実行中は、キー入力を行ってはいけない
- これらのコマンドは、子プロセスとしては実行できない

ただし、これらの制約を無視しても、実際には実行できる場合もありますが、何らかのトラブルの原因になる可能性があるでしょう。

日本語入力システム（FEP）の入れ替え

では、これらの点に注意して、日本語入力システムの「ATOK」と「松茸」を入れ替える実行例を示しましょう。

ATOK6を組み込む

```
A>TYPE ¥ETC¥ATOK.DEV    ……組み込むデバイスドライバ(ここではATOK)を記述したテキストファイル「ATOK.DEV」を見る
DEVICE = A:¥SYS¥ATOK6A.SYS /D=A /G=A
DEVICE = A:¥SYS¥ATOK6B.SYS

A>TYPE ATOK.BAT    ……ATOK組み込み用のバッチファイルの内容
DELDRV
ADDDRV A:¥ETC¥ATOK.DEV    ……DELDRVコマンドを実行した後でADDDRVする

A>ATOK    ……バッチファイルを実行する

A>DELDRV    ……バッチファイルの内容が実行される

A>ADDDRV A:¥ETC¥ATOK.DEV
Microsoft (R) Character Device driver install and deinstall  Version 1.01
Copyright (C) Microsoft Corp 1987. All rights reserved.

自由文変換システム  ＡＴＯＫ６ ver 1.2
Copyright 1986,87 (株)ジャストシステム    ATOK6が組み込まれた

A>    ……CTRL＋XFERキーでATOK6をONにする
⇩
_    これ以降、ATOK6が動作する                連ローマ字漢字
```

ATOK6を松茸に切り替える

```
A>TYPE ¥ETC¥MATUTAKE.DEV ⏎ ……組み込むデバイスドライバ(ここでは松茸V2とマウスドライバ)
                              を記述したテキストファイル「MATUTAKE.DEV」を見る
DEVICE = A:¥SYS¥MOUSEK.DRV
DEVICE = A:¥SYS¥MTTK2.DRV A:¥MATU /J4/T/N/Xｽ/Mｺ/P

A>TYPE MATUTAKE.BAT ⏎ ……松茸V2組み込み用のバッチファイルの内容
DELDRV
ADDDRV A:¥ETC¥MATUTAKE.DEV ……DELDRVコマンドを実行した後で、ADDDRVする

A>MATUTAKE ⏎ ……バッチファイルを実行する

A>DELDRV ……バッチファイルの内容が実行される

A>ADDDRV A:¥ETC¥MATUTAKE.DEV
Microsoft (R) Character Device driver install and deinstall  Version 1.01
Copyright (C) Microsoft Corp 1987. All rights reserved.

MOUSEK: ＭＳ－マウスドライバ (ver2.44) を組込みました.
     (C) 1987 (株)管理工学研究所
---------------------------------------------------------------
日本語入力フロントプロセッサ「 松茸」 Ver 2.04 (C) 1987,88 (株)管理工学研究所
---------------------------------------------------------------
                                    マウスドライバと松茸V2が組み込まれた
A> ……XFERキーで松茸をONにする
            ⇩
      これ以降、松茸が動作する

Rか  A↓  貼込  8:19  記号  複写 松茸ｶﾅ変換 半角変 ﾛｰﾏ字  半角  カナ  Rか
```

図4.32　「ATOK」と「松茸」のデバイスドライバを入れ替える実行例

このようにADDDRV／DELDRVコマンド自体の操作は簡単ですが、ここでの実行例のように、デバイスドライバの入れ替え用のバッチファイルを作っておく方がよいでしょう。

ワープロの切り替え

次に、前項のFEPの入れ替えを応用して、ワープロソフトの一太郎と新松の切り替えを行ってみましょう。もちろんMS-DOSを再起動せずに行うわけです。それぞれのFEPの組み込みには、先の実行例で使ったDEVICE指定のファイルを使います。

では、その一連の実行例を示します。まず最初は、一太郎のユーザーが一太郎を立ち上げます。

```
A>TYPE TOJXW.BAT ⏎ ……………一太郎起動用のバッチファイルの内容を見る
DELDRV                     } ATOK6を組み込む(図4.32参照)
ADDDRV A:¥ETC¥ATOK.DEV     }
A:         } 一太郎Ver.3のあるディレクトリに移動する
CD ¥JXW3   }
JXW ×××× ……………一太郎の起動。××××には、ユーザー各自のパラメータが必要であれば記述する

A>TOJXW ⏎ ……………バッチファイルを実行する
```

↓

```
[1]                [2]                     《 一 太 郎 》    1 頁    1 行    1 列
[1] ■-------10--------20--------30--------40--------50--------60--------70
 |
                        一太郎が起動した
```

図 4.33a　ワープロ一太郎を立ち上げる

このようにすでに起動している MS-DOS 上で、一太郎が立ち上がりました。さて、ワープロ作業を行った後、通常の操作で一太郎を終了し、MS-DOS のプロンプトにもどったとしましょう。

次は新松のユーザーが、新松を立ち上げます。その実行例を示します。

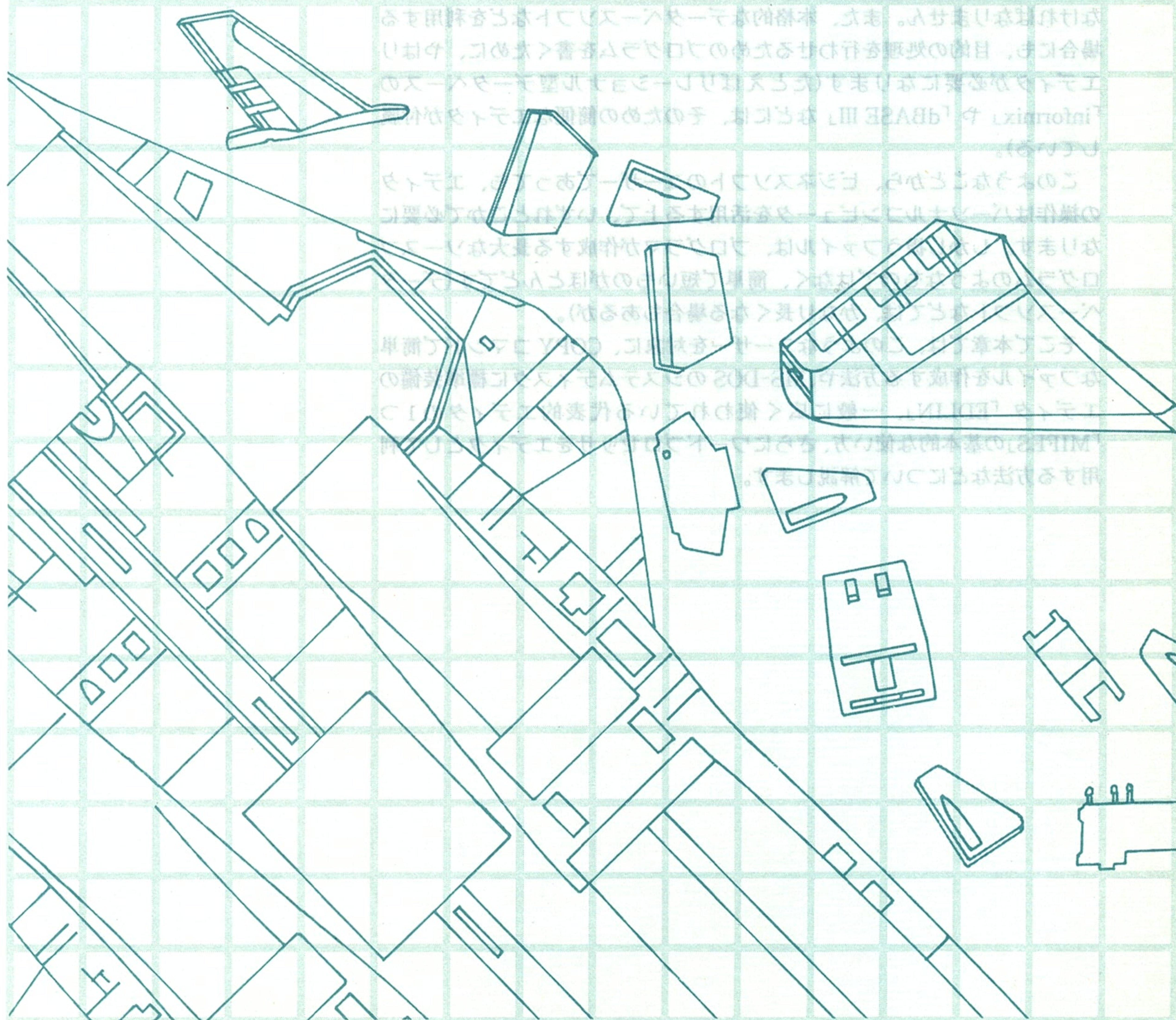
```
A>TYPE TOMATU.BAT ⏎ ……………新松起動用のバッチファイルの内容を見る
DELDRV                        } 松茸V2を組み込む(図4.32参照)
ADDDRV A:¥ETC¥MATUTAKE.DEV    }
A:         } 新松のあるディレクトリに移動する
CD ¥MATU   }
MATU ××××× ……………新松の起動。×××××には、ユーザー各自のパラメータが必要であれば記述する

A>TOMATU ⏎ ……………バッチファイルを実行する
```

↓

```
  日本語ワードプロセッサ　松                 ver 01.03  ser 01311-860094.08
                           新松が起動した   Copyright (C) 1988 ㈱管理工学研究所
```

図 4.33b　ワープロ新松を立ち上げる

このように ADDDRV／DELDRV コマンドを使うことにより、複数の FEP やワープロを、それぞれのユーザーに合わせて自由に切り替えることができます。これが可能なのも、大容量のハードディスクならではのことですが、大勢の人が使うオフィスのマシンなどには、このような配慮がぜひとも必要でしょう。

5章
エディタによる簡単な文書ファイルの作成および編集

本章では、ソフトウェア開発の専門家ではなく、主にビジネスソフトなどを利用する一般的なユーザーを対象に、エディタのやさしい使い方の実習をします。

ソフトウェアを開発するプログラマにとって、エディタは作業時間のもっとも多くを費やす重要なツール(ソフトウェアの道具)です。したがってプログラマの場合は、優れたエディタを選ぶとともに、その操作法に精通しなければなりません。

一方のビジネスソフトのユーザーはどうでしょうか。たとえば、1章や3章などで実習したように、MS-DOSを効果的に利用するには、CONFIG.SYSファイルや、自動スタート・バッチファイルなどの作成／変更を行わなければなりません。また、本格的なデータベースソフトなどを利用する場合にも、目的の処理を行わせるためのプログラムを書くために、やはりエディタが必要になります(たとえばリレーショナル型データベースの「informix」や「dBASE III」などには、そのための簡便なエディタが付属している)。

このようなことから、ビジネスソフトのユーザーであっても、エディタの操作はパーソナルコンピュータを活用する上で、いずれどこかで必要になります。しかし扱うファイルは、プログラマが作成する長大なソースプログラムのようなものではなく、簡単で短いものがほとんどです(データベースソフトなどでは、かなり長くなる場合もあるが)。

そこで本章では、このようなユーザーを対象に、COPYコマンドで簡単なファイルを作成する方法や、MS-DOSのシステムディスクに標準装備のエディタ「EDLIN」、一般に広く使われている代表的エディタの1つ「MIFES」の基本的な使い方、さらにワードプロセッサをエディタとして利用する方法などについて解説します。

5.1 COPYコマンドによるファイルの作成

本書の随所で見られるように、ほんの数行の簡単なテキストファイル(純粋に文字コードだけでできている文字ファイル)を作成する場合には、MS-DOSの内部コマンド「COPY」を使うと便利です。COPYコマンドは本来、「ディスク⇄ディスク」間のファイルのコピーを行うだけでなく、ディスクを含めて、プリンタ、キーボード、ディスプレイ、RS-232Cインターフェイスなどの、各種周辺装置間のファイル転送プログラムです。したがって、「キーボード→ディスク」の転送、つまり、「キー入力データをディスクファイルへ格納する」ことも可能です。

COPYコマンドで私たちが日常使っているのは、次のような「ディスクファイル→ディスクファイル」の転送(コピー)です。

```
A>COPY  x:ファイル名1  y:ファイル名2⏎
```

このコマンドラインは説明の必要もないでしょうが、ドライブx:上のファイル「ファイル名1」を、ドライブy:上に、ファイル名「ファイル名2」としてコピーするものです(同じファイル名でコピーする場合は、「ファイル名2」は不用ですが)。

そこで、このコマンドラインの「x:ファイル名1」を**CON**(コンソール)に変更すれば、キーボードからの入力が、ドライブy:上に「ファイル名2」としてコピーされるはずです。CONは、「コンソール」に割り当てられている**デバイスファイル名**と呼ばれているもので、通常は、キーボードとディスプレイのことを指します。したがって、上のコマンドラインを、

```
A>COPY  CON  y:ファイル名⏎
```

とすれば、キー入力されたものが、任意のファイル名でドライブy:上(y:を省略すればカレントドライブ上)に作成されることになります。

また、COPYコマンドを応用すると、既存ファイルの先頭、または最後に、キーボードからの入力行を追加することができ、さらに、2つ以上の既存ファイルを連結して、1つのファイルにすることも可能です。

■ 1）新しいファイルの作成

■ 3）既存ファイルの先頭にキー入力行を追加

■ 2）既存ファイルの最後にキー入力行を追加

■ 4）複数のファイルの連結

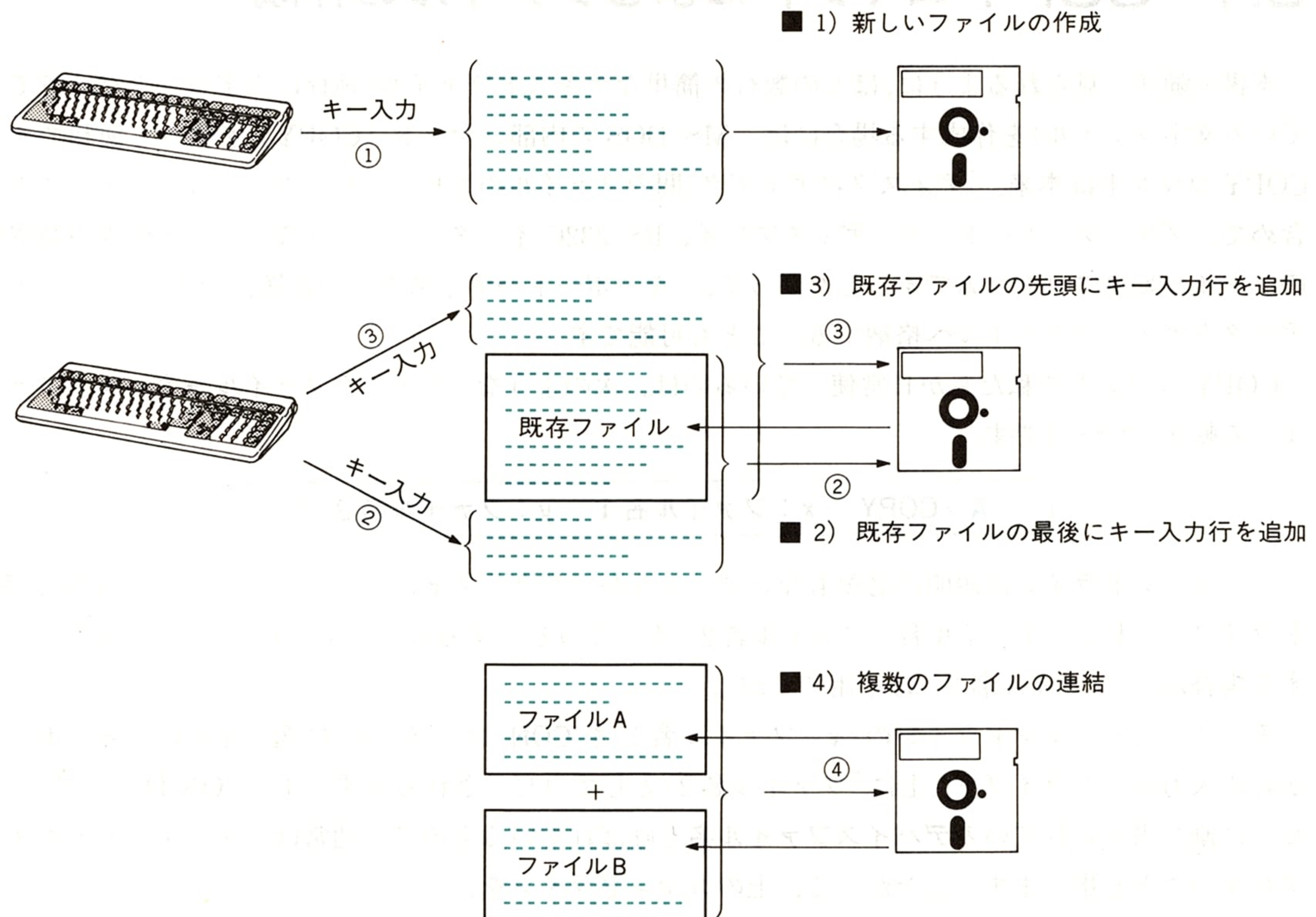



図 5.1　COPY コマンドによるファイルの作成／追加／連結の概念

では、上の**図 5.1** の 1)～4)のそれぞれの場合の実行例を順に示しましょう。いずれも、入力するテキスト(文)の各行の最後には⏎を入力しますが、⏎の入力後は、それまでの入力文字を削除／訂正することはできません。もし⏎の入力後に、どこかを訂正しなければならない場合は、[CTRL]＋[C]をキー入力して COPY コマンドを中止し、コマンドレベルにもどして再度 COPY コマンドの実行から(つまり最初から)やり直しするほかありません。

まず、「1)新しいファイルの作成」の実行例を示します。作成するファイルのファイル名は、適当に「ABCD.CPY」とでもしておきましょう。

```
                    コンソールという「デバイスファイル」(つまりキー入力)をファイル「ABCD.CPY」にコピーすると考える
A>COPY CON ABCD.CPY⏎ ……………COPYコマンドを使って、ファイル「ABCD.CPY」を作成する
Good morning COPY⏎
おはようコピー君⏎
Good afternoon COPY⏎          キー入力したファイルの内容
こんにちはコピー君⏎
Good night COPY⏎
おやすみＣＯＰＹ　また明日⏎
^Z⏎ ……………CTRL+Z ⏎を入力。COPYコマンドを終了させる
        1 個のファイルをコピーしました.

A>DIR ABCD.CPY⏎ ……………作成されたファイルの確認

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

ABCD     CPY      123  89-08-14   19:50 ……………123バイトのファイルが作成されている
        1 個のファイルがあります.          (ファイル中の⏎は2バイトのコード)
   120832 バイトが使用可能です.

A>TYPE ABCD.CPY⏎ ……………作成されたファイルの内容をタイプアウトする
Good morning COPY
おはようコピー君
Good afternoon COPY
こんにちはコピー君
Good night COPY
おやすみＣＯＰＹ　また明日

A>
```

図 5.2 ケース 1) COPY コマンドを使って新しいファイルを作成する

このように新しいファイル「ABCD.CPY」が作成されました。日本語の入力も自由ですので、適当に試みてください。もし、作成するファイルと同名のファイルがすでにディスク上に存在している場合は、もとのファイルは削除され、今回の COPY コマンドで作成されるファイルと置き換わりますので注意してください。

次は「2)既存ファイルの最後にキー入力行を追加」の実行例です。1)で作成したファイル「ABCD.CPY」の最後に追加してみましょう。COPY コマンドの「＋」記号によるファイル連結の機能を利用します。

```
A>COPY ABCD.CPY+CON ⏎ ……………ファイル「ABCD.CPY」の最後にキー入力行を追加する
ABCD.CPY
CON          } COPYコマンドによる表示
BOTTOM ADD ⏎ ……………この1行を追加する
^Z ⏎ ……………CTRL+Z ⏎
        1 個のファイルをコピーしました.

A>TYPE ABCD.CPY ⏎ ……………タイプアウトして内容を確認する
Good morning COPY
おはようコピー君
Good afternoon COPY
こんにちはコピー君          } 既存ファイルの部分
Good night COPY
おやすみCOPY　また明日
BOTTOM ADD ……………キー入力行が最後に追加されている

A>
```

図 5.3　ケース 2）COPY コマンドを使って、既存ファイルの最後へキー入力行を追加する

このように、キー入力した「BOTTOM ADD」の1行が、ファイル「ABCD.CPY」の最後に追加されています。この場合は、もとあったファイル「ABCD.CPY」に直接追加されますが、COPY コマンドを、

```
A>COPY ABCD.CPY+CON ファイル名⏎
```

のように実行すれば、もとのファイル「ABCD.CPY」はそのままの状態で、キー入力行が追加された新しいファイルが、「ファイル名」として作成されます。各自で試みてください。

次は、「3)既存ファイルの先頭にキー入力行を追加」の実行例です。

```
A>COPY CON+ABCD.CPY ABCD1.CPY ⏎ ……………ファイル「ABCD.CPY」の先頭にキー入力行が追加された
                                          ファイル、「ABCD1.CPY」を作成する
CON ……………COPYコマンドによる表示
TOP ADD ⏎ ……………この1行を追加する
^Z ⏎ ……………CTRL+Z ⏎
ABCD.CPY ……………COPYコマンドによる表示
        1 個のファイルをコピーしました.
```

```
A>TYPE ABCD1.CPY⏎ ……………タイプアウトして内容を確認
TOP ADD ……………キー入力行が先頭に追加されている
Good morning COPY
おはようコピー君
Good afternoon COPY
こんにちはコピー君           ｝既存ファイルの部分
Good night COPY
おやすみＣＯＰＹ　また明日
BOTTOM ADD

A>
```

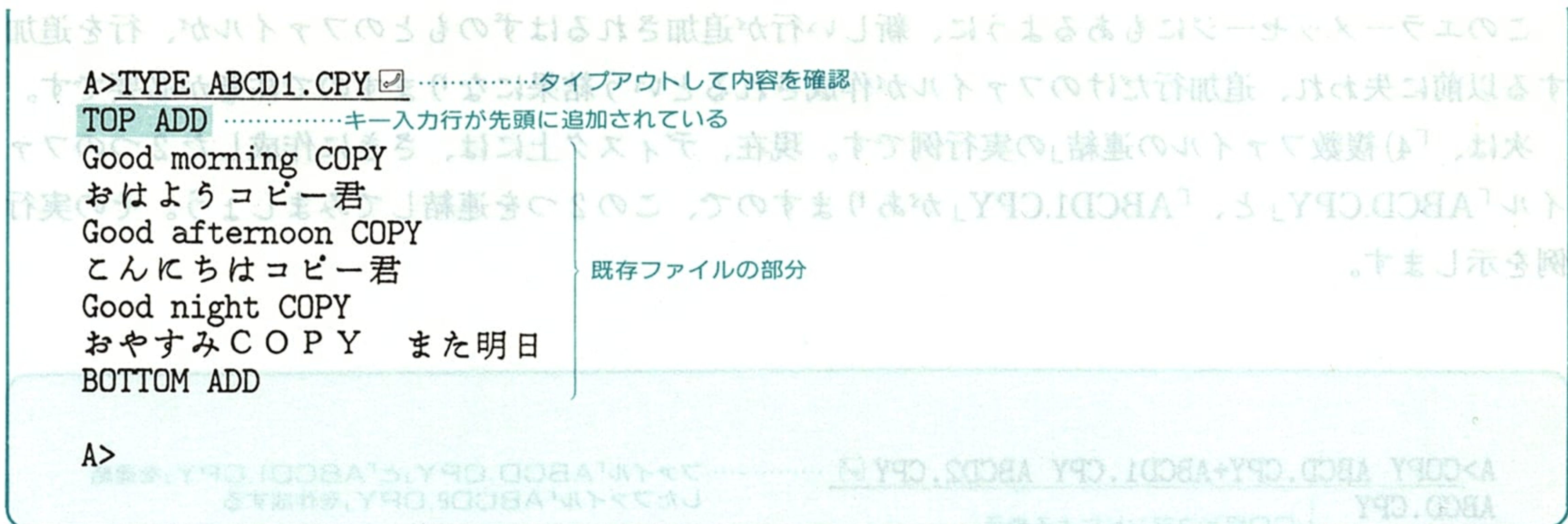



図 5.4 ケース 3) COPY コマンドを使って、既存ファイルの先頭へキー入力行を追加する

このように、「TOP ADD」の１行が、ファイル「ABCD.CPY」の先頭に追加された新しいファイル「ABCD1.CPY」が作成されています。この場合注意することは、COPY コマンドを、次のように実行してはいけません。

A＞COPY CON＋ABCD.CPY ABCD.CPY⏎ (×)

同一ファイル名

試みに、このコマンドを実行してみましょう。

```
A>COPY CON+ABCD.CPY ABCD.CPY⏎ ……………誤ったCOPYコマンドの使い方
CON
TOP ADD⏎ ……………この１行を追加するつもり
^Z⏎ ……………CTRL+Z⏎
ABCD.CPY
コピー前に送り側の内容が失われました.
コピー前に送り側の内容が失われました.
        1 個のファイルをコピーしました.

A>TYPE ABCD.CPY⏎ ……………タイプアウトして内容を確認
TOP ADD ……………この１行のみ存在し、既存ファイルの内容はすべて失われている

A>
```

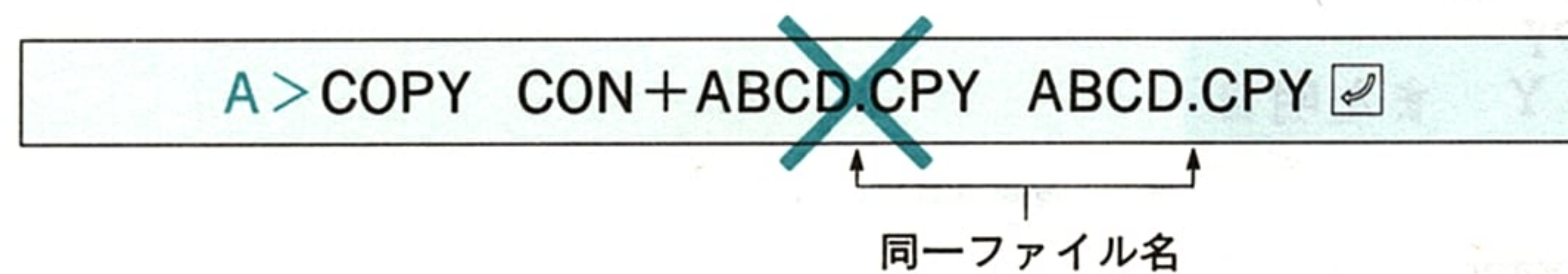
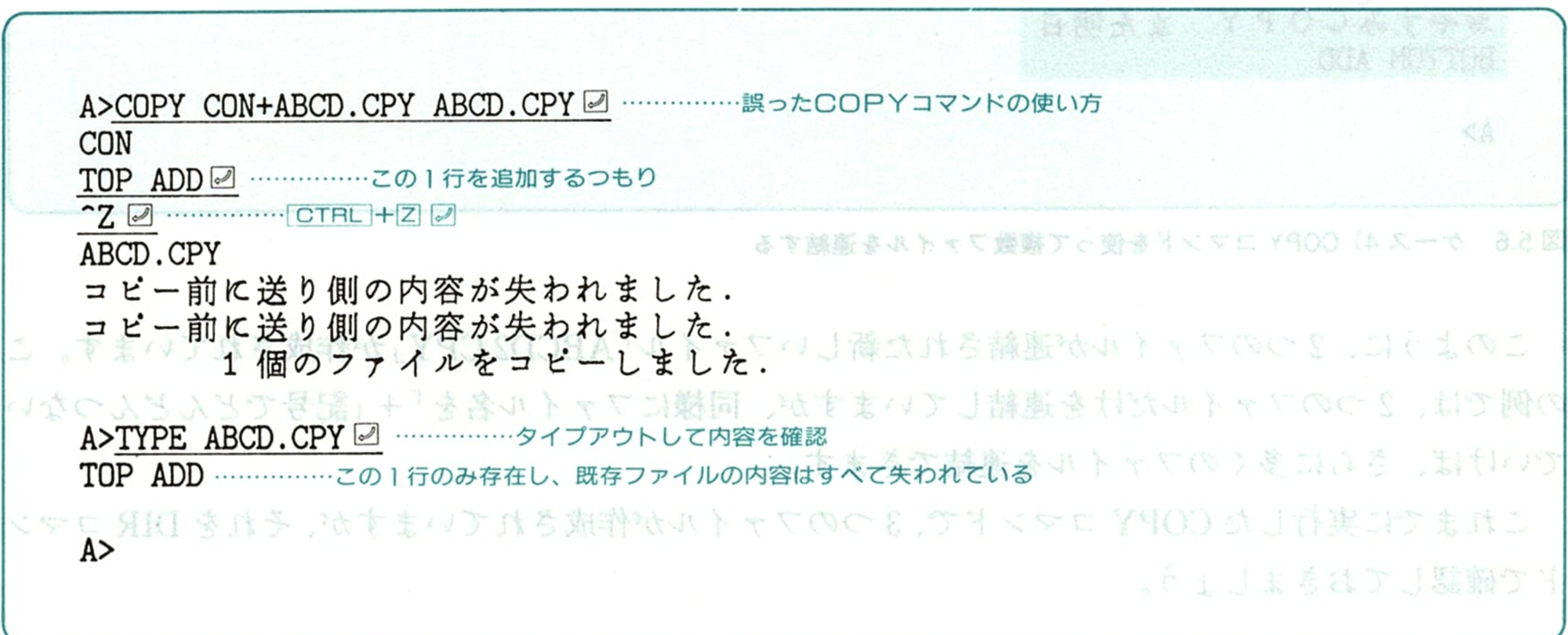



図 5.5 誤った COPY コマンドの使い方

このエラーメッセージにもあるように、新しい行が追加されるはずのもとのファイルが、行を追加する以前に失われ、追加行だけのファイルが作成されるという結果になりますので注意が必要です。

次は、「4)複数ファイルの連結」の実行例です。現在、ディスク上には、さきに作成した2つのファイル「ABCD.CPY」と、「ABCD1.CPY」がありますので、この2つを連結してみましょう。その実行例を示します。

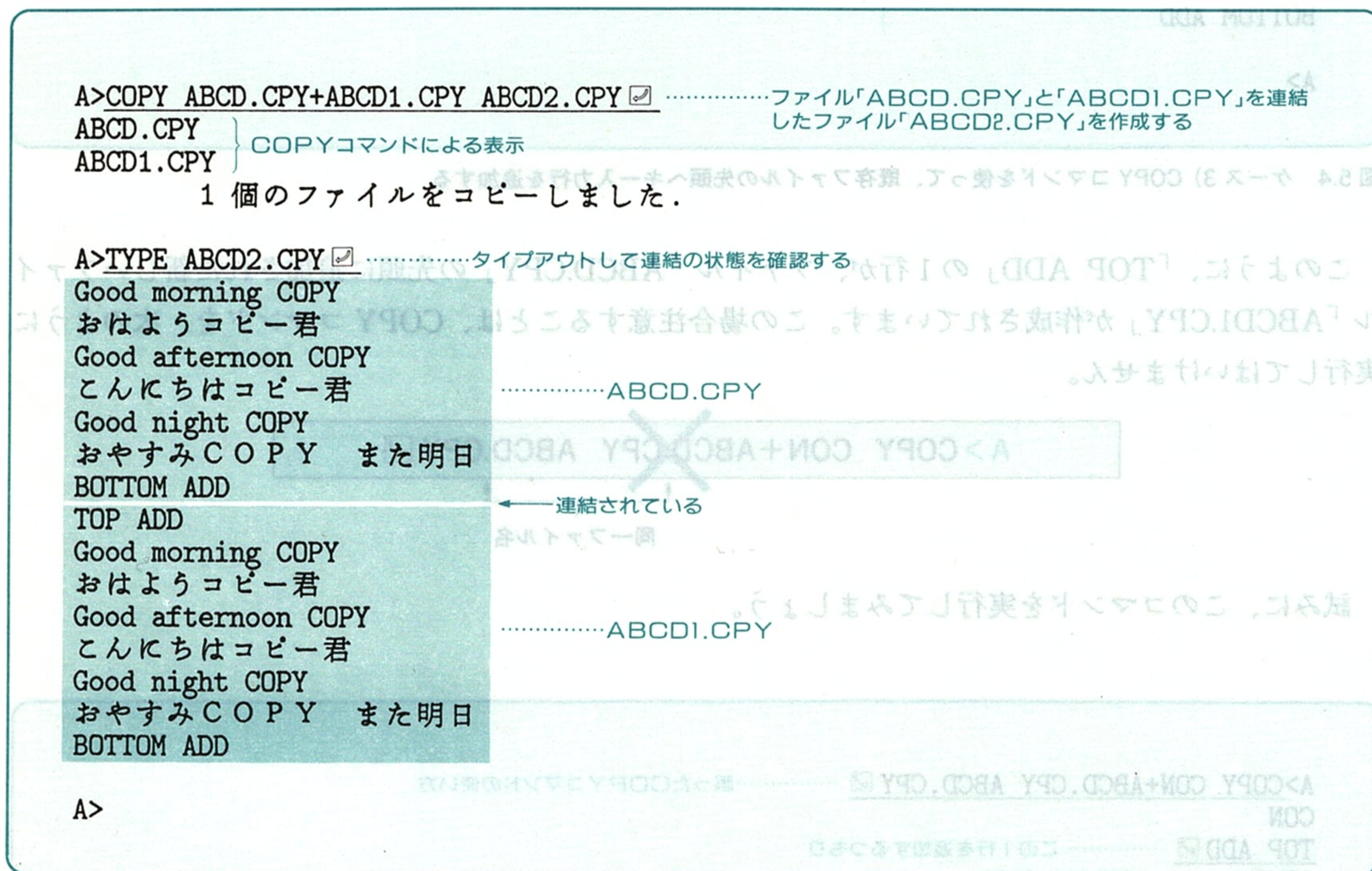
```
A>COPY ABCD.CPY+ABCD1.CPY ABCD2.CPY
ABCD.CPY
ABCD1.CPY
         1 個のファイルをコピーしました.

A>TYPE ABCD2.CPY
Good morning COPY
おはようコピー君
Good afternoon COPY
こんにちはコピー君
Good night COPY
おやすみＣＯＰＹ　また明日
BOTTOM ADD
TOP ADD
Good morning COPY
おはようコピー君
Good afternoon COPY
こんにちはコピー君
Good night COPY
おやすみＣＯＰＹ　また明日
BOTTOM ADD

A>
```


図 5.6　ケース 4) COPY コマンドを使って複数ファイルを連結する

このように、2つのファイルが連結された新しいファイル「ABCD2.CPY」が作成されています。この例では、2つのファイルだけを連結していますが、同様にファイル名を「＋」記号でどんどんつないでいけば、さらに多くのファイルを連結できます。

これまでに実行したCOPYコマンドで、3つのファイルが作成されていますが、それをDIRコマンドで確認しておきましょう。

```
A>DIR *.CPY⏎

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

ABCD      CPY      136  89-08-14   19:53
ABCD1     CPY      145  89-08-14   19:55
ABCD2     CPY      280  89-08-14   20:00
        3 個のファイルがあります.
    113664 バイトが使用可能です.

A>
```

COPYコマンドの実習で、この3つのファイルが作成された

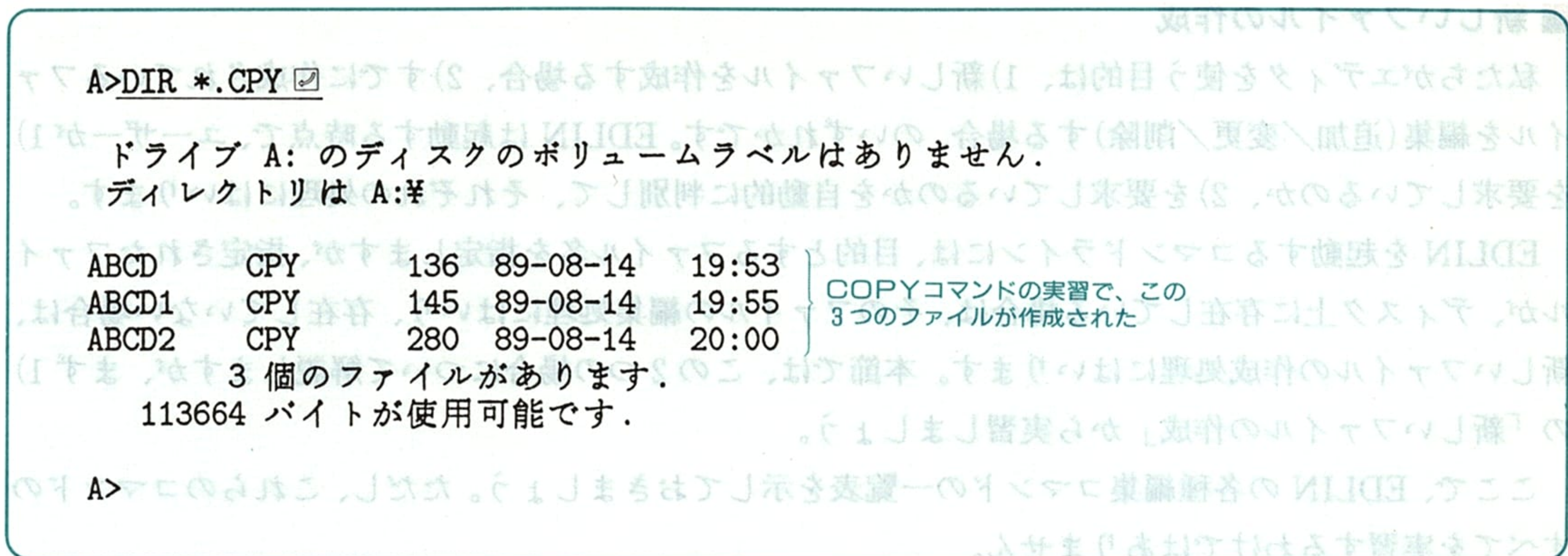



図 5.7　今までの COPY コマンドの実習で作成されたファイルを確認する

以上のように、COPY コマンドを使って、簡単なテキストファイルの作成や、既存ファイルの前後へのキー入力行の追加、それにファイルの連結を行うことができます。エディタを使用するまでもない簡単なテキストファイルの作成などに、手軽に利用してください。

5.2　エディタ「EDLIN」のやさしい使い方

MS-DOS のシステムディスクには、「EDLIN（エドリン）」というエディタが含まれています（プログラムファイル名は「EDLIN.EXE(.COM)」）。EDLIN は、いわゆるスクリーンエディタではありません。現在一般に使われているエディタは、すべて**スクリーンエディタ**――スクリーン上に表示された編集対象のテキスト（文）に対して、スクリーンカーソルを移動させて編集点を決め、画面上で編集作業を行う形式のエディタです。それに対して EDLIN は、画面上に表示された編集対象のテキストに対して、直接編集作業を行うことができない、**ラインエディタ**とか「ポインタ形式」などと呼ばれるエディタです。CRT ディスプレイなどの表示装置が一般化する以前の、紙にプリントアウトされたテキストを対象に編集作業を行っていた時代のエディタであり、今は使われなくなった古いタイプのエディタの典型的な存在です。

本節では、この EDLIN を使って、小さなファイルの作成や編集を行うための、必要最小限の使い方を解説しましょう。プログラマを志す方は、市販されているいくつかのスクリーンエディタのなかから優れたものを購入し、それをマスターする方がよいでしょう。いずれにせよエディタは（ワープロもそうですが）、自在に使いこなすまでには、それ相当の習熟期間が必要です。ただし本節は、そのようなレベルの話ではなく、あくまでやさしい基本的な使い方の実習です。

■ 新しいファイルの作成

私たちがエディタを使う目的は、1)新しいファイルを作成する場合、2)すでに作成されているファイルを編集(追加／変更／削除)する場合、のいずれかです。EDLINは起動する時点で、ユーザーが1)を要求しているのか、2)を要求しているのかを自動的に判別して、それぞれの処理にはいります。

EDLINを起動するコマンドラインには、目的とするファイル名を指定しますが、指定されたファイルが、ディスク上に存在している場合は、そのファイルの編集処理にはいり、存在していない場合は、新しいファイルの作成処理にはいります。本節では、この2つの場合について解説しますが、まず1)の「新しいファイルの作成」から実習しましょう。

ここで、EDLINの各種編集コマンドの一覧表を示しておきましょう。ただし、これらのコマンドのすべてを実習するわけではありません。

コマンド		機能
編集一般	I	インサートモードにはいる。インサートモードの終了は CTRL + C または CTRL + Z ↵
	D	行の削除
	R	文字列の置き替え
	C	行のコピー
	M	行の移動
	S	文字列のサーチ
リストアウト	L	行のリストアウト
	P	ページ単位(23行)のリストアウト
ディスク関係	A	ディスクから編集エリアへ未入力行を読み込む
	W	編集エリアの行をディスクへ格納する
	T	ディスク上の任意のファイルを編集エリアに読み込む
終了	E	EDLINを終了する
	Q	今までの編集作業をすべてキャンセルし、EDLINを終了する

表 5.1　EDLINの各種コマンドの一覧表

新しいファイルを作成するのための EDLIN の起動

エディタ「EDLIN」は、次の書式のコマンドで起動します。これは、1)の場合も、2)の場合も同じです。もちろんディスク上にはEDLINのプログラムファイル「EDLIN.EXE(.COM)」が必要です。

```
A>x：EDLIN　y：ファイル名↵
```

「x：」は、EDLINのプログラムファイル「EDLIN.EXE(.COM)」が存在しているドライブ名であり、「y：」は、新しく作成されるファイル「ファイル名」がセーブ(格納)されるドライブ名です(階層ディレクトリの任意のディレクトリに対しては、そのパス名を付ける)。もし、ドライブy：上に「ファイル名」というファイルがすでに存在している場合は、既存ファイルの編集作業となります。また、x：やy：が省略された場合は、それぞれカレントドライブが指定されます。

新しいファイルを作成する場合のEDLINの働きを次の図で示しましょう。

ファイル「ABCD.EDL」は存在するか
存在していない
EDLINの起動
A>EDLIN ABCD.EDL⏎
(入力編集作業)
メモリ上の編集エリア
キー入力
コンピュータ本体
キー入力されたテキストが、「ABCD.EDL」というファイル名でディスクにセーブされる
新しいファイル「ABCD.EDL」が作成される
EDLINの終了コマンド
＊E⏎
キー入力されたテキスト
キー入力
コンピュータ本体

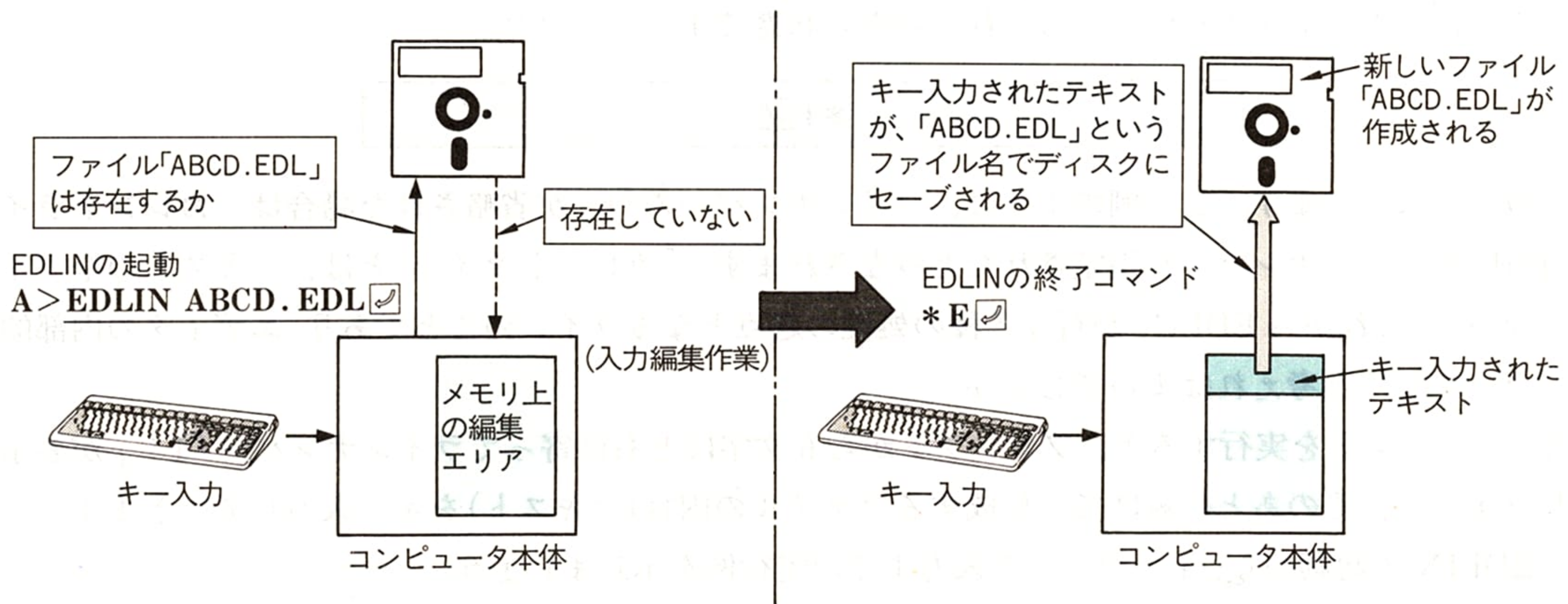



図5.8 新しいファイルを作成する場合のEDLINの働き

では、新しいファイルを作成してみましょう。作成するファイルのファイル名は、現在のディスク上に存在していない名前、たとえば「ABCD.EDL」とします。その場合のEDLINを起動するコマンドの一例を次に示します。

```
A>EDLIN  ABCD.EDL⏎
```

このコマンドラインの意味は、(この場合のカレントドライブはA：なので)「ドライブA：上のEDLINを起動して、ドライブA：上に新しいファイル「ABCD.EDL」を作成する」ということになります(「.EDL」の意味は、EDLINで作成したことがわかるように付けたもの)。

このコマンドラインを実行し、EDLINが起動すると、メッセージに続いて画面の左端に「*」が表示されます。この「*」はEDLINのプロンプトであり、各種の編集操作を行うためのコマンドの受付けが可能な状態であることを示しています。さあ、これ以降はEDLINの世界です。

テキスト(文章)の入力

EDLINが起動したら、「Iコマンド」によりインサートモードにはいります。行の挿入コマンドであるI(Insert)コマンドの基本的な書式を次に示します。

＊nl ⏎	**現在のテキストのラインナンバーnと、その上の行との間に新しい行を追加する。つまり、挿入されたあとは、nが挿入した行のラインナンバーとなる(ラインナンバーは、1から最大65533までの整数)**

ここではテキストがまだ何も入力されていない状態ですので、ただ、

＊I ⏎

とのみキー入力します。この例のように、ラインナンバーの「n」が省略された場合は、**カレントライン**(後述)のラインナンバーが指定されたとみなされます。「カレントライン」とは、テキストの全ラインのなかで、これからEDLINが行う各種の処理の起点となるラインのことであり、エディタの内部的な〝カーソル〟と考えればよいでしょう。

このIコマンドを実行すると、プロンプトから6文字ほど右に寄ってラインナンバー「1：」が表示されますので、そのあとに続けて、作成するファイルの内容(テキスト)をキー入力していきます。では、EDLINの起動から、Iコマンドの入力までの実行例を示しましょう。

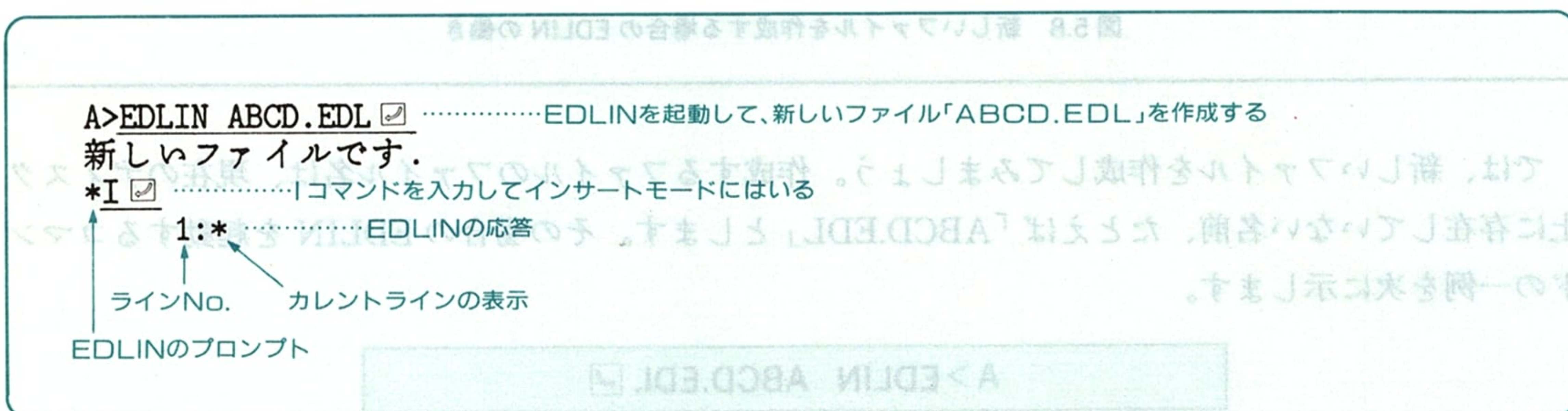
```
A>EDLIN ABCD.EDL ⏎   ……EDLINを起動して、新しいファイル「ABCD.EDL」を作成する
新しいファイルです.
*I ⏎                 ……Iコマンドを入力してインサートモードにはいる
      1:*             ……EDLINの応答
```


図5.9　新しいファイルを作成するためにEDLINを起動し、インサートコマンド「I」を実行する

この実行例からもわかるように、EDLINのプロンプトが表示される位置は最左端ですが、各コマンドに対するEDLINの応答や、テキストの入力などは、6文字ほど右に寄った位置から始まります。つまり、各種コマンドの入力ができるのは、最左端の「＊」であり、テキストの入力位置はそれより6文字右側ですので、両者を区別することができます。画面上のこの位置の違いに注目してください。

Iコマンドを実行すると、この場合は「1：＊」と表示されますが、この「1：」の意味は「キー入力がラインナンバー1として挿入される」ことを示し、そのあとの「＊」(EDLINのプロンプトではない)

の意味は、その行――つまりラインナンバー1が「カレントライン」(現在の編集対象行)であることを示しています。

さて、この「1：*」の後ろに、適当なテキストを入力していきます。各行の最後は必ず⏎を入力します。⏎が入力されるたびにラインナンバーが+1され、それにしたがってカレントラインの表示「*」も移っていきます。

インサートモードの終了

Iコマンドを実行し、インサートモードにはいった状態で、テキストを入力していきますが、このインサートモードを終了するには、CTRL + C を入力するか(そのあとの⏎の入力は不要)、あるいは CTRL + Z を入力した後、⏎を入力します。いずれも行の先頭で入力しなければなりません。これによってインサートモードが終了し、EDLINのプロンプトにもどります。

EDLINの終了

新しいファイルの作成や編集作業を終え、処理済みのテキストをディスクにセーブして、EDLINを終了するには、E(End)コマンドを実行します。その書式を次に示します。

```
*E⏎
```

このEコマンドを実行すると、作成したテキストは、新しいファイルの作成であれば、EDLINの起動時に指定したファイル名で、ディスクにセーブされます。また、既存ファイルの編集であれば、もとのファイルの中身はそのままの状態で、ファイルタイプ部(拡張子)の名前が「xxxx.BAK」とリネーム(変更)されてバックアップファイルとして残され、編集済みのテキストがもとのファイル名でディスクにセーブされます。これらのことは図5.8および図5.11に示してありますので参照してください。

もし、行った作業をすべてキャンセルし、ディスク上のファイルをまったく変更せずに、EDLINを終了する場合は、Q(Quit)コマンドを、「*Q⏎」のように実行します。

では、EDLINの起動から、テキストを入力し、EDLINを終了するまでの実行例を通して次ページの図5.10に示しましょう。

```
A>EDLIN ABCD.EDL⏎
新しいファイルです.          前述
*I
      1:*Good morning EDLIN⏎
      2:*おはようエドリン君⏎
      3:*Good aftrnoon EDLIN⏎
      4:*こんにちわエドリン君⏎          テキストを入力していく
      5:*Good night EDLIN⏎
      6:*おやすみＥＤＬＩＮ　また明日⏎
      7:*^Z⏎ ……………CTRL+Z ⏎でインサートモードを終了する(CTRL+Cでもよい)
*E⏎ …………EDLINの終了コマンドEを入力する

A>DIR ABCD.EDL⏎ …………EDLINを終了し、MS-DOSにもどったのでDIRコマンドを
                          実行して作成されたファイルを確認する

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

ABCD     EDL      132  89-08-14   20:30 ……………132バイトのファイルが作成されている
          1 個のファイルがあります.
    103425 バイトが使用可能です.

A>TYPE ABCD.EDL⏎ ……………その内容をタイプアウトして確認する
Good morning EDLIN
おはようエドリン君
Good aftrnoon EDLIN
こんにちわエドリン君          入力したテキストによるファイルが作成されている！
Good night EDLIN
おやすみＥＤＬＩＮ　また明日
A>
```

図 5.10　EDLIN で新しいファイルを作成する一連の実行例

DIR コマンドや TYPE コマンドで確認できるような、新しいテキストファイルが作成されました。この実行例は、I(Insert) コマンドと E(End) コマンドのみを使って、簡単なファイルを作成しただけに過ぎません。このテキストには、故意に 2 箇所の入力ミスがありますが、これらは次に解説する各種編集機能で訂正しましょう。

■ 既存ファイルの編集(追加／変更／削除)

前項では新しいファイルの作成を行いましたが、本項では既存ファイルの編集を行います。この場合も EDLIN の起動は前項と同様です。

A＞x：EDLIN　y：ファイル名⏎

ただし、今回指定するファイルの「ファイル名」は、すでにディスク上に存在しているわけです。既存ファイルを編集する場合の EDLIN の働きを次の図で示しましょう。

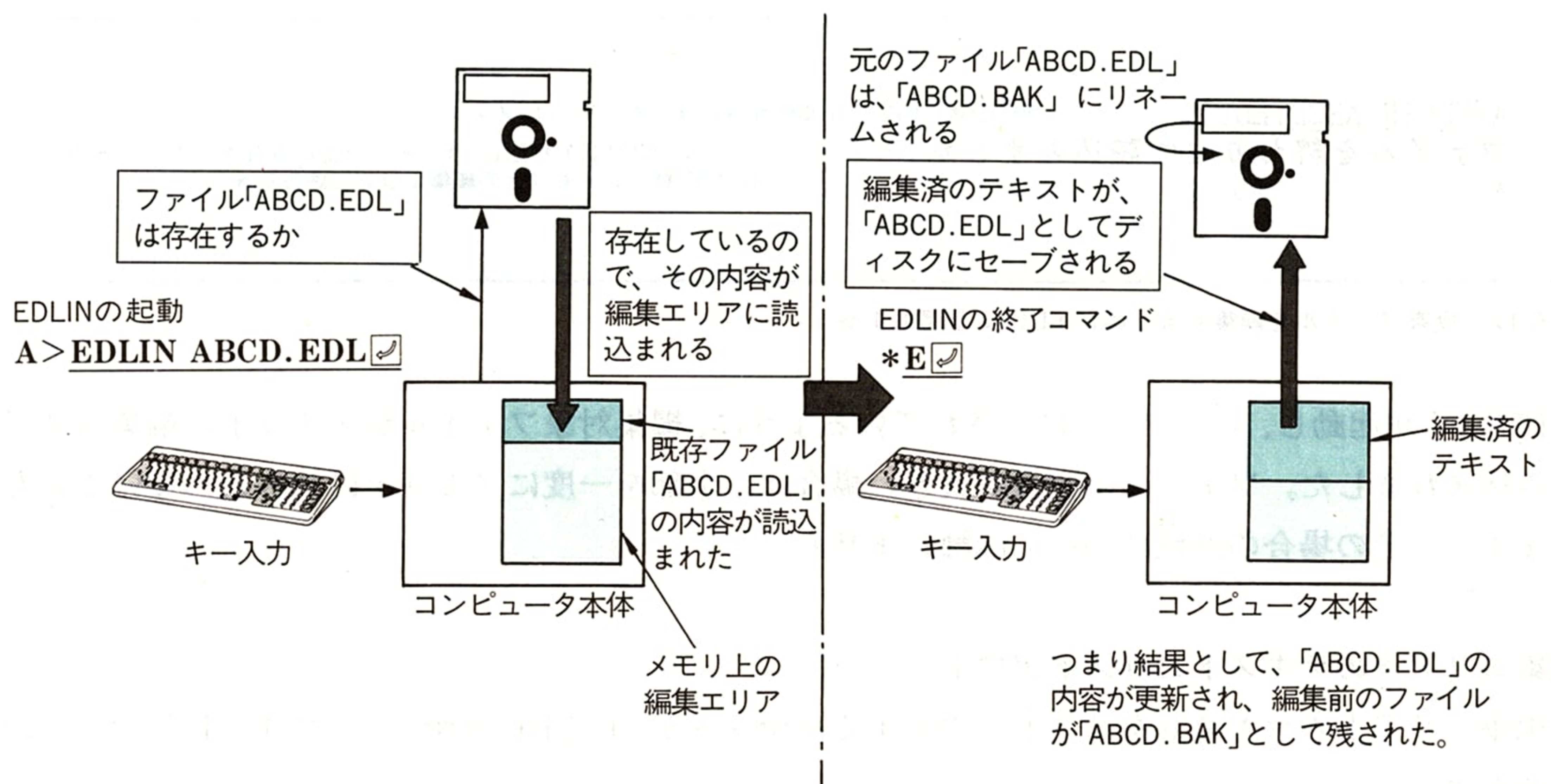



図 5.11　既存ファイルを編集する場合の EDLIN の働き

EDLIN の起動と編集対象ファイルの読込み

上に示したコマンドラインが実行され、EDLIN が起動すると、コマンドラインで指定されているファイル名がディスク上に存在するかどうかの確認が行われます。もし存在していればその内容が自動的にコンピュータのメモリの編集エリアに読み込まれ、編集作業が可能となります。**編集エリア**とは、テキストの各種の編集作業が行われるメモリ上のワークエリア(作業場所)のことをいいます。

では、既存ファイル「ABCD.EDL」(前項で作成したもの)を編集する場合の、EDLIN を起動するコマンドの一例を次に示します。

A＞EDLIN　ABCD.EDL⏎

これは、前項で行った、新しいファイルを作成する場合とまったく同じコマンドラインですが、今回は、「ドライブA：上(カレントドライブ)のEDLINを起動して、ドライブA：上の既存ファイルABCD.EDLを編集する」という意味になります。

このコマンドを実行し、EDLINが起動すれば、ファイル「ABCD.EDL」の内容が自動的にコンピュータのメモリに読み込まれ、各種の編集作業が可能になります。ではまず、ここまでの実行例を示しましょう。

```
A>EDLIN ABCD.EDL⏎ ……………前回の新しいファイルの作成とまったく同じコマンド
ファイルを終わりまで読込みました. ……………ファイル「ABCD.EDL」がディスク上に存在するので、その
                                              内容が自動的にメモリ上の編集エリアに読み込まれた
* ……………EDLINのプロンプト
```

図5.12　既存ファイルを編集するためにEDLINを起動する

EDLINが起動し、メッセージに示されているように、編集対象ファイルがメモリ上の編集エリアに読み込まれました。ファイルの容量が大きい場合は、全部を一度にメモリ上に読み込めないこともありますが、その場合の処理については触れません。

編集エリア上のテキストのリストアウト

編集エリア上のテキストをリストアウトするコマンドが「L(List)コマンド」です。Lコマンドの基本的な書式は、

＊n, mL⏎　　n＝リストアウト開始ラインナンバー、
m＝終了ラインナンバー

ですが、ここではテキスト全体をリストアウトするための、次のようなLコマンドを実行します。

＊1, #L⏎

ラインナンバーに「#」記号を使用した場合、#の意味は「編集エリアの最終ラインナンバー＋1」とみなされます。したがってこのコマンドは、ラインナンバー1から、最後のラインナンバーまでがリストアウトされます。また、「＊1, 100L⏎」のように、最終ラインナンバーより大きい値を設定した場合も、最終ラインナンバーとみなされます。

では、ここまでの実行例を示しましょう。

```
A>EDLIN ABCD.EDL
ファイルを終わりまで読込みました.
*1,#L              ……………Lコマンドで、ラインNo.1から最終ラインまでをリストアウトする
      1:*Good morning EDLIN
      2: おはようエドリン君
      3: Good aftrnoon EDLIN          編集エリア上のすべてのテキストがリストアウト
      4: こんにちわエドリン君           される。ここでは6行のみのテキストである
      5: Good night EDLIN
      6: おやすみＥＤＬＩＮ　また明日
*
```

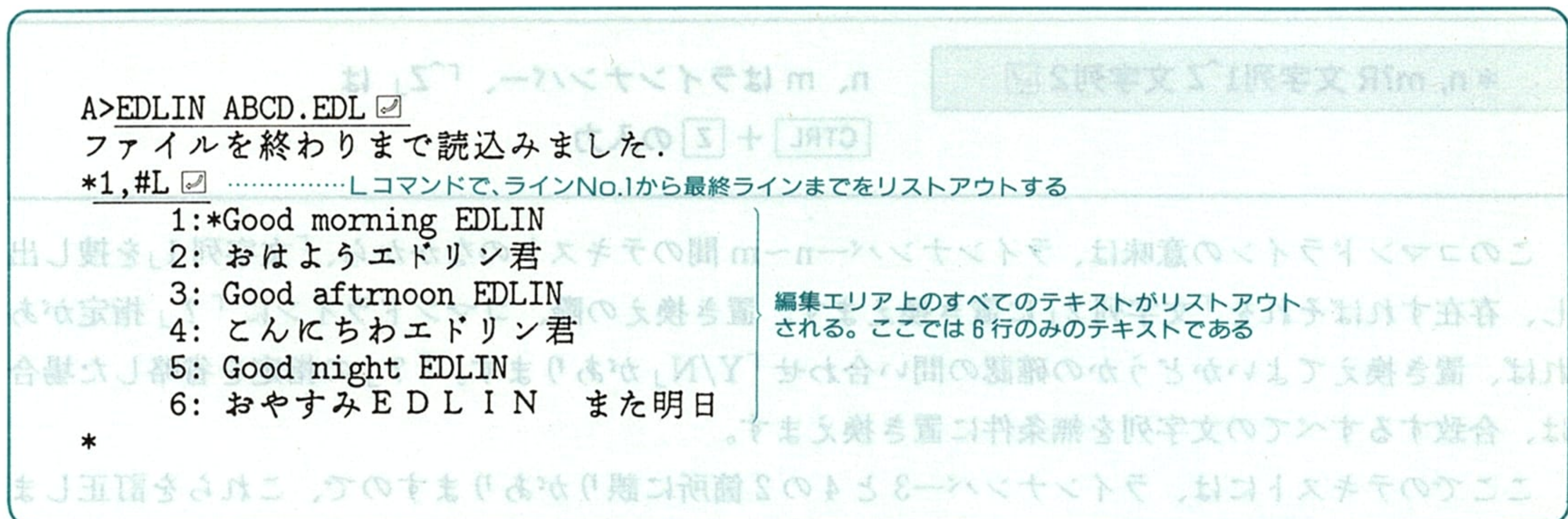
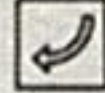



図 5.13　リストコマンド(L)によりテキスト全体をリストアウトする

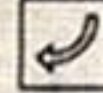
また、図 5.16 の実行例にも示されているように、L コマンドには、ラインナンバーを指定しないで、ただ「＊L」と実行した場合には、カレントラインを中心に前後 11 行、計 23 行がリストアウトされます。上に示した実行例では、カレントラインがラインナンバー1 であり、カレントラインの前に 11 行分のテキストがないので、その分、後方の表示ライン数が増えます。この形式の L コマンドは便利ですから、よく使うことになるでしょう。

前項でも述べましたが、コマンドを入力する位置は最左端のプロンプト「*」であり、EDLIN の応答は 6 文字ほど右に寄った位置から始まっていることに注目してください。また、ラインナンバー1 のテキストの頭に「*」が付いていますが、これは、ラインナンバー1 がカレントラインであることを示しています。

さて編集エリアには、ファイル「ABCD.EDL」が読み込まれ、そのテキスト全体がリストアウトされています。これで編集作業を行うための準備が整いました。ではまず、入力ミスした 2 つの箇所の訂正から始めましょう。

文字／文字列の訂正

テキスト中の文字や文字列を訂正するには、次の方法があります。

- 訂正する箇所が含まれている行全体を、新しい挿入行として再度正しくキー入力し、旧行を削除する（I コマンドと D コマンドを使う）
- テンプレート機能による行編集を行う（この機能には触れない）
- 文字列の置き換えコマンド「R」を利用する

ここでは、誰もが容易に使えて便利な、文字列の置き換えコマンド「R」(Replace)を利用します。Rコマンドの基本的な書式を次に示します。

＊n, m?R 文字列1^Z 文字列2 ⏎	n、m はラインナンバー、「^Z」は CTRL ＋ Z の入力

このコマンドラインの意味は、ラインナンバーn〜m間のテキストのなかから、「文字列1」を捜し出し、存在すればそれを「文字列2」に置き換えます。置き換えの際、コマンドラインに「?」指定があれば、置き換えてよいかどうかの確認の問い合わせ「Y/N」があります。「?」の指定を省略した場合は、合致するすべての文字列を無条件に置き換えます。

ここでのテキストには、ラインナンバー3と4の2箇所に誤りがありますので、これらを訂正しましょう。まずラインナンバー3の「aftrnoon」の綴りの「e」が抜けている誤りは、次のいずれかのRコマンドを実行します。

1) ＊3, 3?Raftr^Zafter ⏎

2) ＊1, #?Raftr^Zafter ⏎

1)の例はラインナンバー3のみを対象にしていますが、2)の例はテキスト全体を対象にしています(#については、前述のLコマンドを参照)。このテキストには、文字列「aftr」は1箇所しかないので実行結果は同じになります。なお、「文字列1」の記述は、この例のように誤りの部分が特定できるように、誤りの箇所の前後の適当な範囲の文字列を指定してください。また、いちいち置き換えるか否かの問い合わせの必要がない場合は、「?」の記述を省略します。

次に、ラインナンバー4の「こんにちわ」の部分の「わ→は」の訂正も、同様にRコマンドで行います。

3) ＊4, 4Rわ^Zは ⏎

4) ＊1, #Rこんにちわ^Zこんにちは ⏎

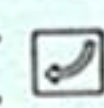
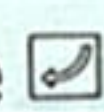
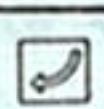
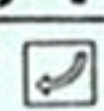
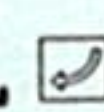
今回は「?」を省略していますので、確認の問い合わせはありません。ではこれらの実行例を示します。

```
A>EDLIN ABCD.EDL⏎
ファイルを終わりまで読込みました.
*1,#L
     1:*Good morning EDLIN
     2: おはようエドリン君
     3: Good aftrnoon EDLIN
     4: こんにちわエドリン君
     5: Good night EDLIN
     6: おやすみＥＤＬＩＮ　また明日
*1,#?Raftr^Zafter⏎ ……………Rコマンドですべての行を対象に、「after」の「e」を挿入する。「Y／N」の問い合せ付き
     3: Good afternoon EDLIN ……………その結果が示される
よろしいですか <Y/N>? Y ……………このように変更してよいなら「Y」を入力する
*4,4Rちわ^Zちは⏎ ……………Rコマンドで、ラインNo.4のみを対象に、「わ」を「は」に変更する。「Y／N」の問い合わせはない
     4: こんにちはエドリン君 ……………変更されている
*1,#L⏎ ……………Lコマンドですべての行をリストアウトする
     1: Good morning EDLIN
     2: おはようエドリン君
     3: Good afternoon EDLIN  ┐
     4:*こんにちはエドリン君  ┘誤りが訂正されている
     5: Good night EDLIN
     6: おやすみＥＤＬＩＮ　また明日
*
```

図 5.14　テキストの誤り箇所を訂正する

新しい行の挿入

テキストの任意の行に、新しい行を挿入するには、前述のI(Insert)コマンドを使用します。前回は、空の編集エリアに新しいテキストを入力しましたが、今回は、すでにあるテキストに新しい行を追加挿入します。

ラインナンバー4と5の間に、

Insert line
挿入行です

の2行を挿入してみましょう。

次ページに示すように、Iコマンドを実行するとEDLINの応答があり、挿入行となるラインナンバーと、「*」が示されますので、この後に続けてテキストを入力していきます。さきに述べたように、ここで表示される「5：*」の「*」は、EDLINのプロンプトではなく(EDLINのプロンプトは画面の最左端に表示される)、カレントラインを示すマークであり、ラインナンバー5がカレントラインであることを示しています(カレントラインの移動については後述)。

＊5I⏎　　………(キー入力)
　5：＊　………(EDLINの応答)

入力するテキストの各行の終わりには⏎を入力します。この⏎の入力のたびに、次に入力される行のラインナンバーが+1されて示されます。同時にカレントラインも移動します。この実行例を示しましょう。

```
*5I⏎ ……………Iコマンドでライン No.5の直前に行を挿入する
     5:*Insert line⏎  }
     6:*挿入行です⏎    } この2行を挿入する
     7:*^C⏎ ……………CTRL+C でインサートモードを終了する

*1,100L⏎ ……………ライン No.1～100の間のテキストをリストアウトする
     1: Good morning EDLIN
     2: おはようエドリン君
     3: Good afternoon EDLIN
     4: こんにちはエドリン君
     5: Insert line  }
     6: 挿入行です    } 挿入された行
     7:*Good night EDLIN
     8: おやすみＥＤＬＩＮ　また明日
*    └──ライン No.は付け直されている
```

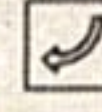



図 5.15　新しい行を挿入する

なお、テキストの最後に新しい行を追加する場合は、すでに述べた「#」記号を使い、次のようにIコマンドを実行します。

＊#I⏎

ラインナンバーに「#」記号を使った場合は、テキストの最終ラインナンバー+1とみなされますので、テキストの最後に新しい入力行が追加されることになります。

行の削除

テキストの任意の行を削除するには、D(Delete)コマンドを使用します。Dコマンドの基本的な書式を次に示します。

＊n, mD⏎

このコマンドラインの意味は、ラインナンバーn～mの範囲のテキストを削除するものです。たとえばラインナンバー2と3の2行を削除する場合のコマンドは、

```
*2, 3D↵
```

となります。この実行例を示しましょう。

```
*2,3D↵ ……………前リストのラインNo.2～3の2行をDコマンドで削除する
*L↵ ……………Lコマンドでカレントラインの前後、計23行をリストアウトする
    1: Good morning EDLIN
    -------------------- ←――この間にあった2行が削除されている
    2:*こんにちはエドリン君
    3: Insert line
    4: 挿入行です
    5: Good night EDLIN
    6: おやすみＥＤＬＩＮ　また明日
*
```

図5.16　テキスト行を削除する

なお、ラインナンバーnから、テキストの最後までを削除する場合は、前項と同じように「#」記号を使い、

```
*n, #D↵
```

とします。

行のコピー

任意の行(複数行も可)のテキストを、任意のラインナンバーの位置にコピーするにはC(Copy)コマンドを使用します。Cコマンドの基本的な書式を次に示します。

```
*n, m, dC↵
```

このコマンドラインの意味は、ラインナンバーn～mの間のテキストを、ラインナンバーdの直前——つまり、ラインナンバーdと、その上の行との間にコピーするものです。たとえば、ラインナンバー2と3の2行を、ラインナンバー5と6の間にコピーする場合のCコマンドは、

```
*2, 3, 6C↵
```

となります。この実行例を示しましょう。

```
*2,3,6C⏎ ……………Cコマンドで前リストのラインNo.2～3の2行をラインNo.6の直前にコピーする
*1,#L⏎ ……………すべてをリストアウト
     1: Good morning EDLIN
     2: こんにちはエドリン君
     3: Insert line
     4: 挿入行です                 コピーされている
     5: Good night EDLIN
     6:*こんにちはエドリン君
     7: Insert line
     8: おやすみＥＤＬＩＮ　また明日
*
```

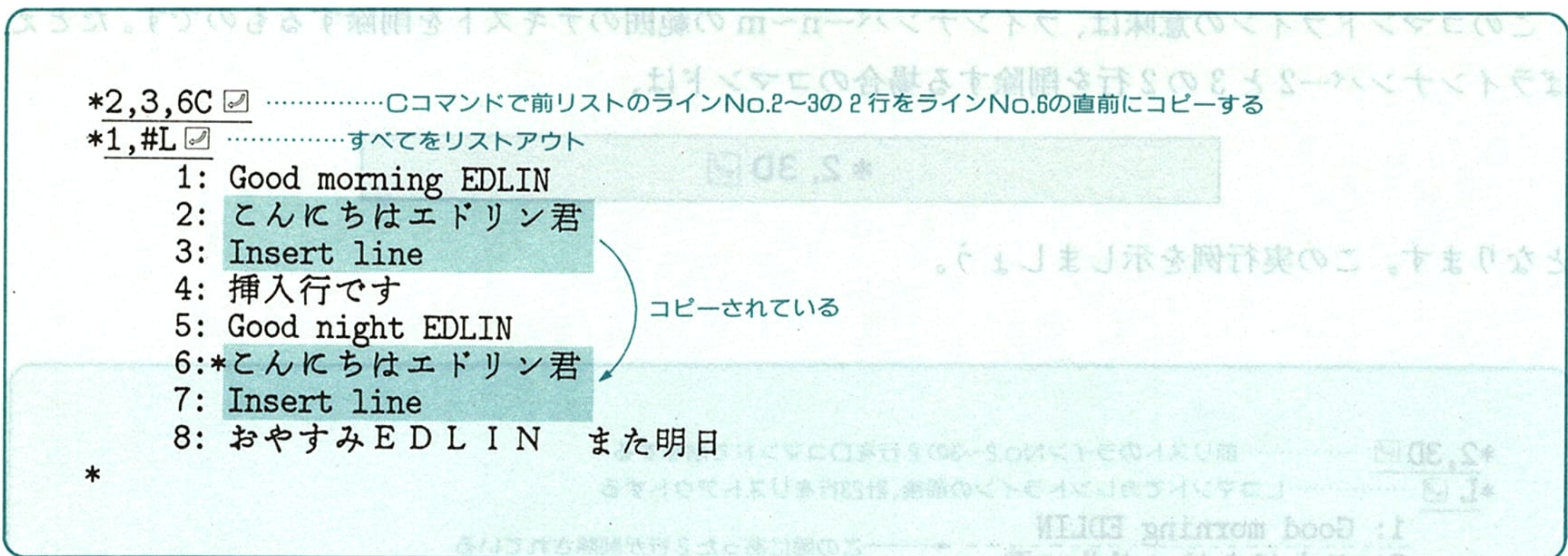



図 5.17　テキスト行をコピーする

なお、任意の1行のみをコピーする場合、たとえば上の実行例で、ラインナンバー3のみをラインナンバー5と6の間にコピーするには、

*3, 3, 6C

とします。

EDLIN の終了

すでに述べたように編集作業が終わった後、処理済みのテキストをディスクにセーブしEDLINを終了するには、Eコマンドを実行します。今回は既存ファイルの編集なので、Eコマンドを実行すると、もとのファイル「ABCD.EDL」は、「ABCD.BAK」というファイル名に変更され、バックアップファイルとしてそのまま残されます。編集作業を受けたテキストは、もとのファイル名「ABCD.EDL」としてディスクにセーブされます。もし、編集したテキストをすべてキャンセルし、EDLINを起動する前の状態にもどすには、Qコマンドを実行します(前述)。

では、EDLINを終了し、編集されたファイルを確認してみましょう。

```
*E↵ ……………Eコマンドで編集エリアのテキストをディスクにセーブして、EDLINを終了し、MS-DOSにもどる

A>DIR ABCD.*↵ ……………編集後のファイルを確認する

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

ABCD     BAK      132  89-08-14   20:41 ……………もとのファイルは「.BAK」とリネームされ、
                                          バックアップファイルとして残されている
ABCD     EDL      151  89-08-14   20:57 ……………編集済のファイル
        2 個のファイルがあります.
    94208 バイトが使用可能です.

A>TYPE ABCD.EDL↵ ……………編集済のファイルをタイプアウトして確認する
Good morning EDLIN
こんにちはエドリン君
Insert line
挿入行です
Good night EDLIN                 編集後のファイル
こんにちはエドリン君
Insert line
おやすみＥＤＬＩＮ　また明日

A>TYPE ABCD.BAK↵ ……………バックアップファイル(元のファイル)をタイプアウトして確認する
Good morning EDLIN
おはようエドリン君
Good aftrnoon EDLIN              編集前のファイル
こんにちわエドリン君
Good night EDLIN
おやすみＥＤＬＩＮ　また明日

A>
```

図 5.18 既存ファイルの編集作業を終了する

このように、編集前のファイルは「.BAK」ファイルとして残され、編集後のファイルがもとのファイル名で作られて、EDLIN による編集作業が無事終わりました。

■ カレントラインの移動

最後に、カレントラインを任意のラインに移動する方法を示しておきましょう。カレントラインは、各種のコマンドの実行にしたがって、自動的に移動しますが、任意の行に自由に移動させることもできます。いずれの場合もカレントラインは、画面上のテキストの、ラインナンバーの後に付けられた「*」表示によって示されます。

任意の行へのカレントラインの移動は、次の実行例のように、ラインナンバーのみを入力して↵した後、もう1度↵します。2度↵することに注意してください。1度目の↵では、テンプレート機能による行編集モード(本書では触れていない)にはいっていますので、ここでキー入力を行うと、もとの行が書き換えられてしまいます。十分に注意してください。

```
*L↵ ……………リストアウト
      1:*Good morning EDLIN ……………現在、この行がカレントライン
      2: こんにちはエドリン君
      3: Insert line
      4: 挿入行です
      5: Good night EDLIN
      6: こんにちはエドリン君
      7: Insert line
      8: おやすみＥＤＬＩＮ　また明日
*5↵ ……………5行目にカレントラインを移動する
      5:*Good night EDLIN
      5:* ………………………………この状態はテンプレートによる編集モードになっている。テンプレートの操作に
                                 慣れていない人は、ただちに↵を入力すること！
*L↵ ……………リストアウト
      1: Good morning EDLIN
      2: こんにちはエドリン君
      3: Insert line
      4: 挿入行です
      5:*Good night EDLIN ……………カレントラインがここに移動した
      6: こんにちはエドリン君
      7: Insert line
      8: おやすみＥＤＬＩＮ　また明日
*
```

図5.19　カレントラインを任意のラインに移動する

以上で、EDLINの初歩的な解説は一通り終わりです。しかし、EDLINを本格的に使うユーザーは、そのほかのコマンドの使い方はもちろん、各コマンドを能率的に使うための操作方法や、そのバリエーションに精通しなければなりません。しかし現在では、本格的な作業には、ほかの優れたスクリーンエディタを使うことになるでしょう。

5.3 スクリーンエディタ「MIFES」のやさしい使い方

ここでは、現在一般に広く使われているエディタの代表の1つとして、「MIFES(マイフェス)」(メガソフト社)を取り上げます。当然ながらMIFESは、前節で紹介したMS-DOSのシステムディスクに標準装備のポインタ形式のエディタ(ラインエディタ)、「EDLIN」とは形態が異なる**スクリーンエディタ**です。

スクリーンエディタとは、本章5.2の冒頭でも触れましたが、画面上に表示されたテキストに対して、スクリーンカーソルを移動して編集箇所を指定し、挿入や削除などの各種の編集作業を、画面上で自由に行うことができる形態のエディタです。EDLINなどのラインエディタに比べ、直感的でわかりやすく能率的であるため、現在の「エディタ」は、ほとんどすべてがこのような「スクリーンエディタ」です。

本節では、そのスクリーンエディタの1つであるMIFESを使って、小さなファイルの作成や編集を行うための、必要最小限の使い方を解説しましょう。

■ MIFESを使うための最低限の心得

簡単なファイルを作成したり編集したりするのであれば、次のことがわかっていれば、最低限のことは行うことができます。

MIFESを起動し、既存ファイルの編集であればそのファイルを読み込む。

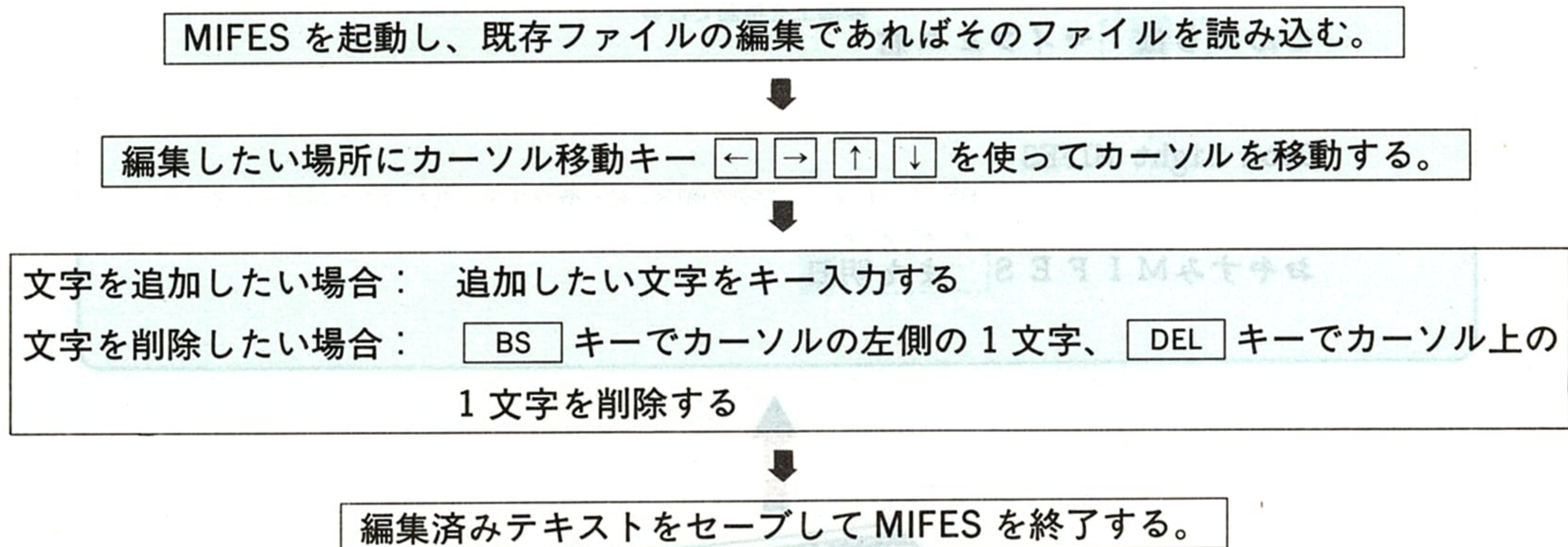
↓

編集したい場所にカーソル移動キー [←] [→] [↑] [↓] を使ってカーソルを移動する。

↓

文字を追加したい場合：　追加したい文字をキー入力する
文字を削除したい場合：　[BS] キーでカーソルの左側の1文字、[DEL] キーでカーソル上の1文字を削除する

↓

編集済みテキストをセーブしてMIFESを終了する。

MIFESを使ったテキストの作成／編集作業は、いちおうこれだけで最低限のことは行うことができます。MIFESには、当然、多種多様の便利な機能がたくさん備っていますが、それらは必要に応じて、おいおいと覚えていけばよいでしょう。本項では、必要最低限の機能のほかにも、いくつかの便利な機能を紹介しますが、スクリーンエディタにおける編集作業の基本は、「カーソルの移動と文字の追加／削除にある」ということを忘れないでください。

MIFESを起動する操作

MIFESを終了する操作

テキストファイル

テキストファイル

編集するファイルを読み込む

作成したファイルをディスクにセーブする

Good morning MIFES

文字列の追加

挿入された文字数分、既存文字が右にずれる

おはよう，元気かな　マイフェス君

Good afternoon MIFES

カーソル移動

↑、↓、←、→キーで自由に画面上を移動できる

こんにちは　マイフェス君

Good night MIFES

BSキーによる文字列の削除。続く既存文字があれば左へ詰められる

おやすみＭＩＦＥＳ　また明日

テキストの入力と編集作業

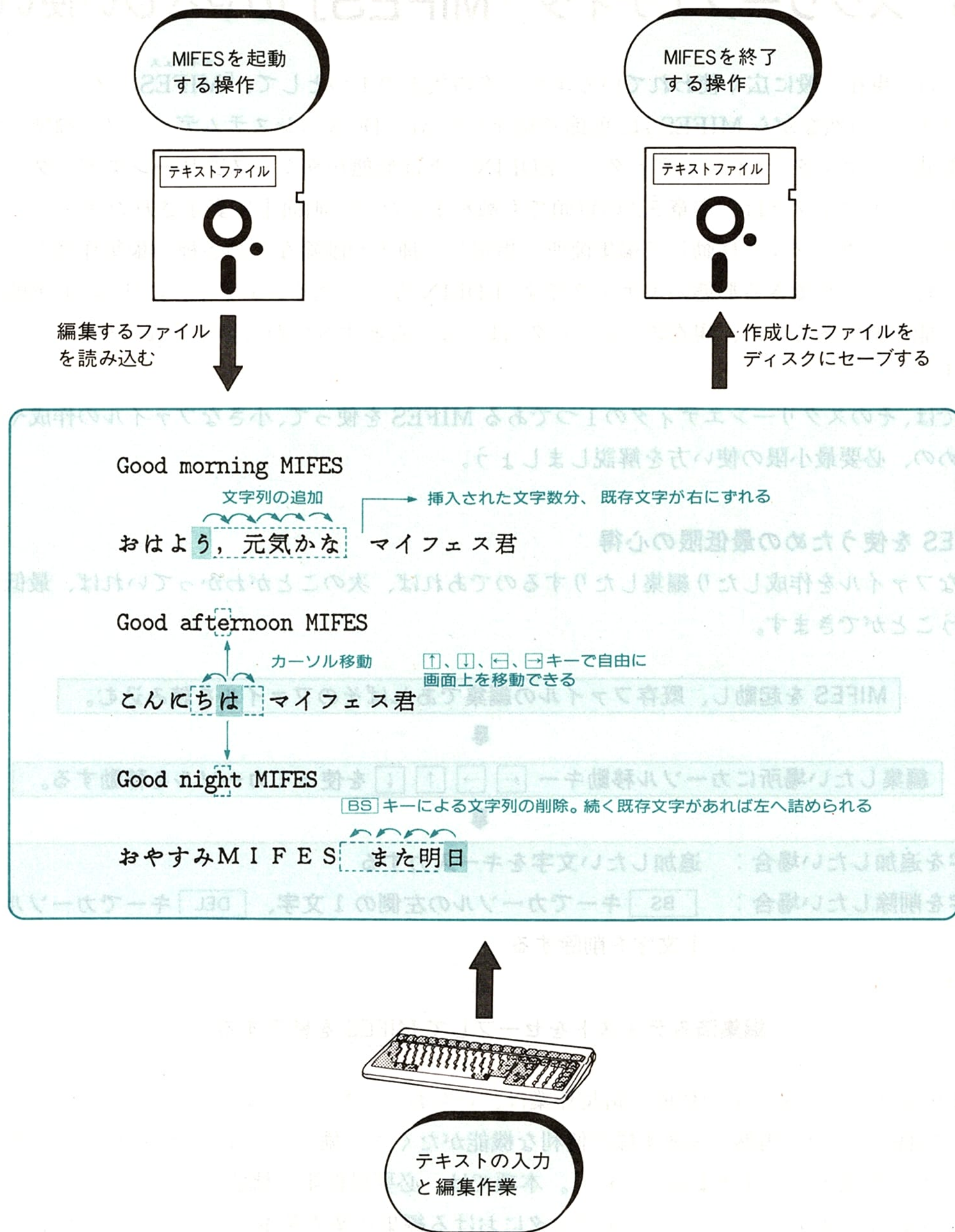



図5.20　スクリーンエディタにおける編集作業の基本

■ MIFES を使うための準備と MIFES の起動

MIFESのオリジナルディスクに含まれている各種のファイルを次に示します。このなかで、MIFESが動作するために最低限必要なファイルは、MIFES本体のプログラムファイル「MIFES.EXE」だけです。したがって、本節でMIFESを実習するには、「MIFES.EXE」だけを、使用する各自のディスクにコピーしておけばよいわけです。

```
A>dir⏎

 ドライブ A: のディスクのボリュームラベルは MIF9841
 ディレクトリは A:¥

CONFIG   SYS      52  88-09-03   4:13
AUTOEXEC BAT     116  88-09-03   4:13
MIFES    EXE   98424  89-07-19   4:13
MIFES    HLP   42412  89-05-09   4:13
MIKEY    EXE   91400  89-04-29   4:13
MIFIND   EXE    7821  89-03-17   4:13
MITAGS   EXE   17820  88-09-03   4:13
MIFORM   EXE   41864  88-09-03   4:13
MIPP     EXE   22942  88-09-03   4:13
MILIB    EXE   12986  89-04-06   4:13
TABCONV  EXE   14330  88-09-03   4:13
MIDEL    EXE   10644  88-09-03   4:13
MILC     EXE   17729  88-09-03   4:13
MILOGO   COM    3015  88-09-03   4:13
MI       BAT      42  88-09-03   4:13
MIA      BAT      79  88-01-11   4:13
MIB      BAT      78  88-01-11   4:13
MIKEY    DEF   16402  89-03-15   4:13
MIKEYSMP DEF   17582  88-09-03   4:13
MIKEY_C  DEF   17157  88-09-03   4:13
MIKEY_WS DEF   16967  88-09-03   4:13
README   DOC   20873  88-10-05   4:13
MIBACKUP BAT     617  88-10-06   4:13
SAMPLE   DOC   10007  88-09-03   4:13
MIFORM   [illegible]  88-09-03   4:13
[illegible] MIL  834  89-04-0[illegible]
FILTER   MIL     560  89-04-06   4:13
CHILDWIN MIL     563  89-04-06   4:13
MIFES    DOC   52067  88-08-18   4:13
MIFES    LIB   26892  89-04-06   4:13
MIWS     EXE   98424  89-05-09   4:13
        53 個のファイルがあります.
      516096 バイトが使用可能です.

A>
```


図 5.21　MIFES のオリジナルディスクに含まれる各種のファイル

エディタ「MIFES」は、次の書式のコマンドで起動します。新規ファイルの作成の場合も、既存ファイルの編集の場合も同じです。もちろんディスク上にはMIFESのプログラムファイル「MIFES.EXE」が必要です。

```
A>x：MIFES　y：ファイル名↵
```

「x：」は、MIFESのプログラムファイル「MIFES.EXE」が存在しているドライブ名であり、「y：」は、新しく作成されるファイル「ファイル名」がセーブ(格納)されるドライブ名、あるいは編集しようとする既存ファイル「ファイル名」が存在するドライブ名です(階層ディレクトリの任意のディレクトリに対しては、そのパス名を付ける)。ドライブy：上に「ファイル名」というファイルが存在していない場合には、新しいファイルの作成となり、存在している場合は、既存ファイルの編集となります。このことは、前節のEDLINと同じですので、図5.8(223ページ)および図5.11(227ページ)を参照してください。また、x：やy：が省略された場合は、それぞれカレントドライブが指定されます。

では、MIFESのプログラムファイル(MIFES.EXE)の準備ができたら、さっそく簡単なファイルを作成してみましょう。

■ 新しいファイルの作成

新しいファイルを作成するには、前節のEDLINと同様に、MIFESを起動するためのコマンドラインにおいて、作成するファイル名を指定します。ここでは、カレントドライブ(A：)上のMIFESを起動して、同じドライブ(A：)上に新規ファイルを作成しましょう。次のコマンドを入力します。

```
A>MIFES　WXYZ.MIF↵
```

作成するファイルのファイル名は「WXYZ.MIF」としておきました(「.MIF」の意味は、MIFESで作成したことがわかるように付けたもの)。MIFESが起動すると次のような画面が表示されます。

A:¥WXYZ.MIF　新しいテキストです<挿入>　HELPでヘルプ
[EOF]
~
~　ファイルの終わりを意味するマーク
~
~
~
~
~
~
~
~
~
~
~
~
~
~
MENU1　MENU2　MENU3　SPLIT　FIND　SEL　CUT　COPY　PASTE　TAGJP

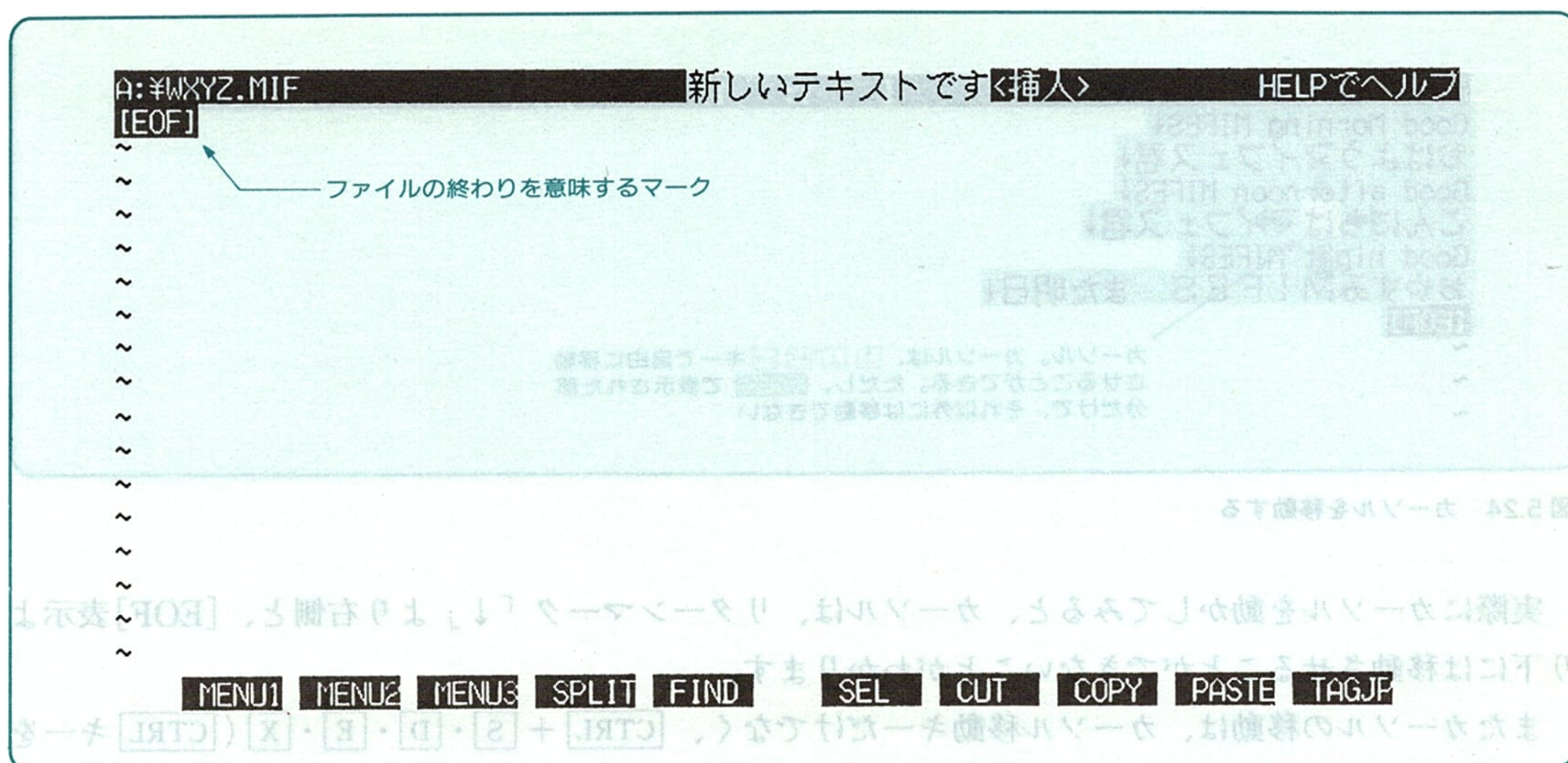



図 5.22　MIFES が起動した直後の画面

この画面には、まだテキストが何も入力されていないため、カーソルを移動することはできません。[EOF]という表示は、ファイルの終わり(End Of File)を意味しています。

ではこの画面に対して、次のようにテキストを入力していきましょう。EDLIN と違って、テキストを入力するための挿入コマンドなどは必要ありません。そのままどんどんキー入力していけば、入力した文字は画面のカーソルの位置に書き込まれていきます。

A:¥WXYZ.MIF　[　7:　1]　<挿入>　HELPでヘルプ
Good Morning MIFES↓
おはようマイフェス君↓
Good afternoon MIFES↓
こんにちはマイフェス君↓　リターンを入力したところには、「↓」マークが黄色で表示される
Good night MIFES↓
おやすみＭＩＦＥＳ　また明日↓
[EOF]
~
~
~　次々にキー入力していく

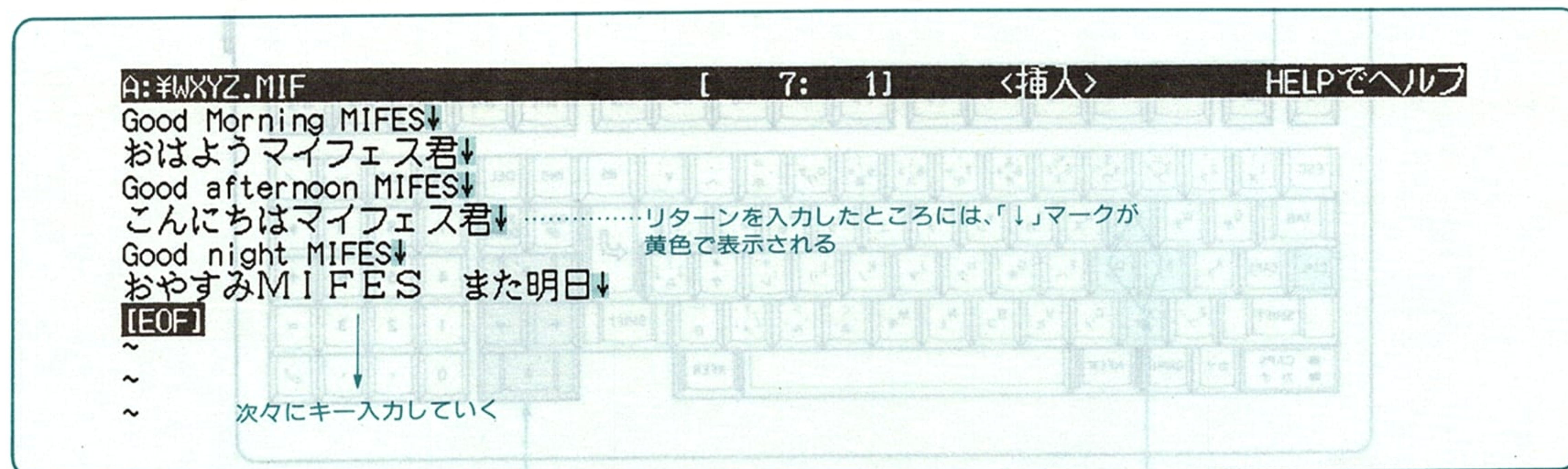



図 5.23　テキストを入力していく

リターンキーを入力したところには、「↓」のマークが黄色で表示されます。カーソルの位置は、現在、[EOF]の表示のところにあり点滅しています。ここでちょっとカーソルを移動してみましょう。カーソル移動キー ← → ↑ ↓ を使って、自在に移動することができます。

```
A:¥WXYZ.MIF                    [     7:    1]      <挿入>          HELPでヘルプ
Good Morning MIFES↓
おはようマイフェス君↓
Good afternoon MIFES↓
こんにちはマイフェス君↓
Good night MIFES↓
おやすみＭＩＦＥＳ　また明日↓
[EOF]
~
~
~
```

カーソル。カーソルは、↑ ↓ ← →キーで自由に移動させることができる。ただし、　　で表示された部分だけで、それ以外には移動できない

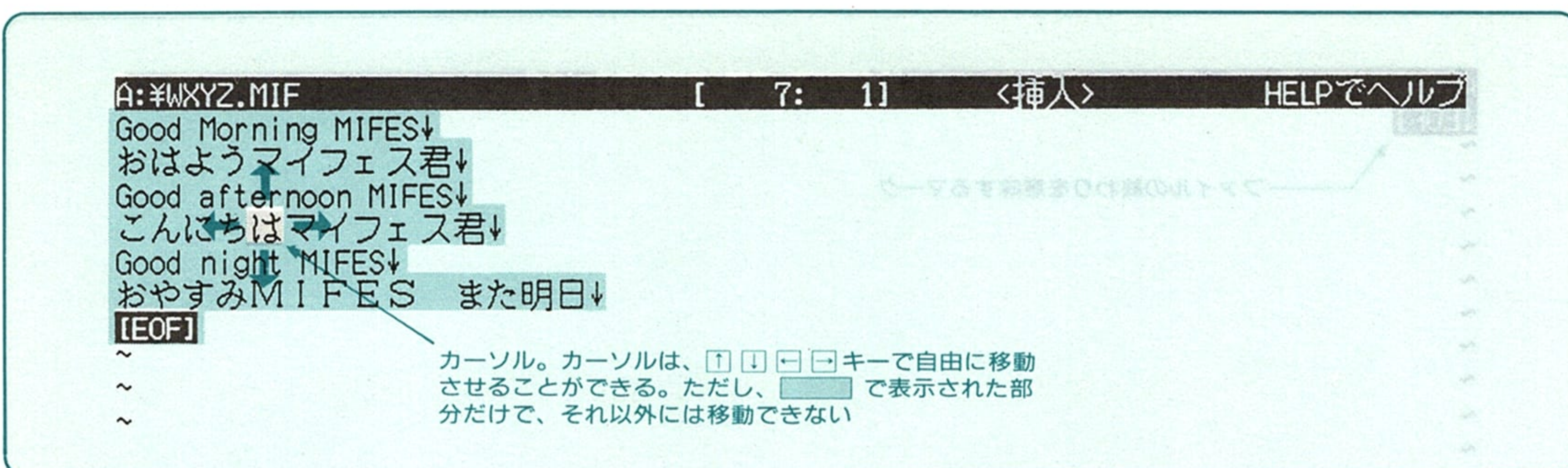



図 5.24　カーソルを移動する

実際にカーソルを動かしてみると、カーソルは、リターンマーク「↓」より右側と、[EOF]表示より下には移動させることができないことがわかります。

またカーソルの移動は、カーソル移動キーだけでなく、CTRL＋S・D・E・X（CTRLキーを押しながらS、D、E、Xの各キーを押す）によっても行うことができます。これらは**ダイヤモンドカーソル**と呼ばれ（キーの配置が菱形の格好をしているため）、キーボードのホームポジションから手を離さずにカーソルを移動できるので、ブラインドタッチができる人は、こちらの方が、格段にスピーディなカーソル移動が行えます。なお、このダイヤモンドカーソルは、米国での代表的なワープロソフトであるWORDSTARや、スクリーンエディタの元祖WORDMASTERを始め、内外の多くのエディタやワープロに採用されています（ワープロ「新松」や「一太郎」にも採用されている）。

ダイヤモンドカーソル

CTRL ＋ ←S　D→　↑E　X↓

カーソル移動

カーソル移動キーを使う場合は、ホームポジションから手を離さなければならない

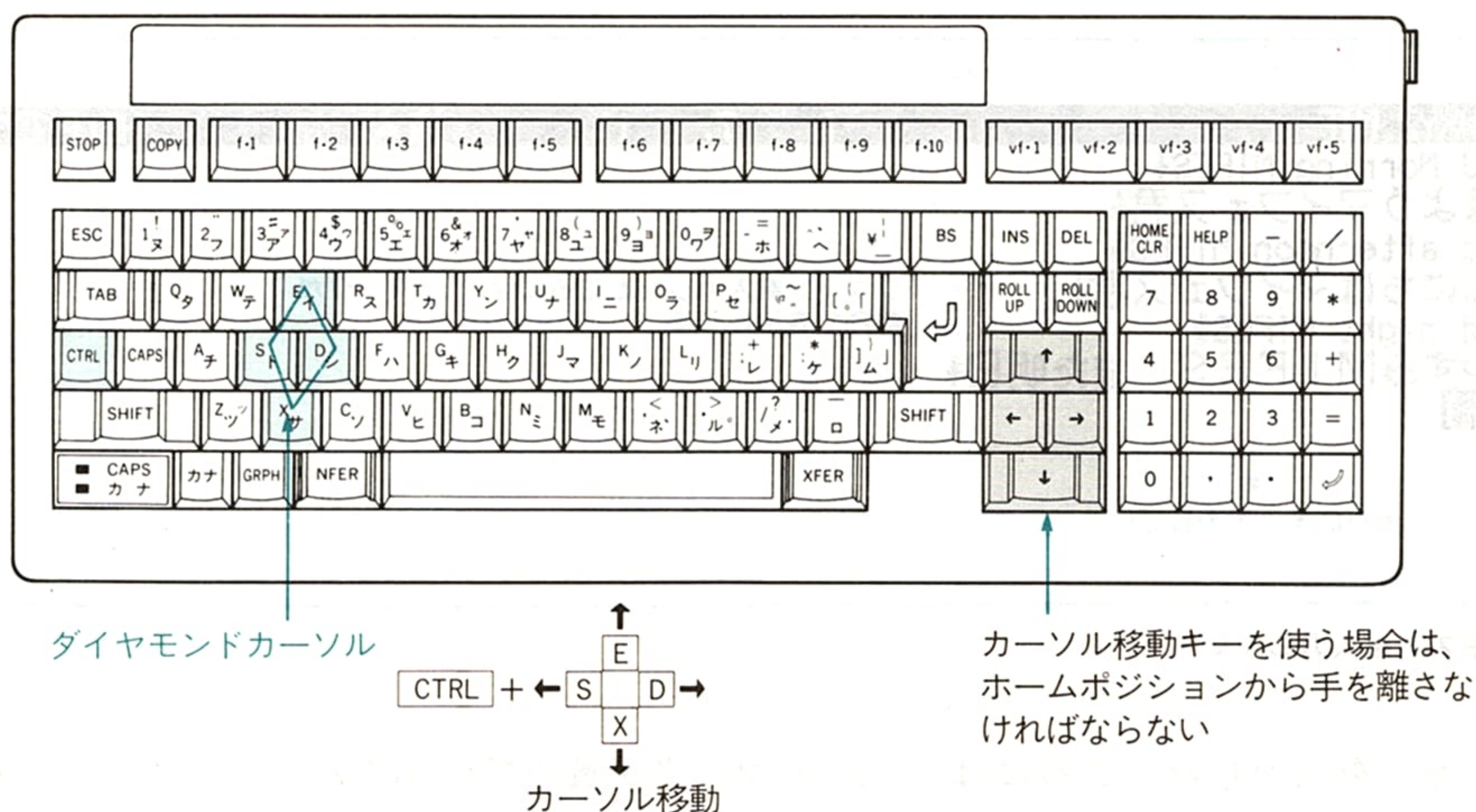



図 5.25　スピーディなカーソル移動が可能なダイヤモンドカーソル

さて、テキストの入力が終わったら、次はそれをディスクにセーブします。MIFESでは、各種のコマンドがファンクションキーに割り当てられていますので、まず f･1 を入力して「ファイル関係の操作」のメニューを表示させます(そのままもとの編集画面にもどすには、何も入力せずに↵するか、あるいは ESC キーを押します)。

```
A:¥WXYZ.MIF              [    7:    1]      <挿入>           HELPでヘルプ
Good Morning MIFES↓
おはようマイフェス君↓
Good afternoon MIFES↓
こんにちはマイフェス君↓
Good night MIFES↓
おやすみＭＩＦＥＳ　また明日↓
      ファイル関係の操作
E Save & Quit    :現テキストのセーブと編集終了
S Save           :現テキストのセーブ　編集は継続
Q Quit           :現テキストの強制的な編集終了
O Original file  :現テキストの編集を最初からやり直す
D Duplicate      :現テキストの二重化
R Rename         :現テキストのパス名の変更
A All Save & End:全テキストのセーブとエディタ終了
0 End            :強制的なエディタの終了
I Insert file    :カーソル行の前にファイルを読み込む
N New file       :新しいファイルの編集
~
~
~
~
~
 MENU1  MENU2  MENU3  SPLIT  FIND    SEL  CUT  COPY  PASTE  TAGJP
```

ここでは「E」を選びテキストをセーブした後、MIFESを終了する

メニューの各項目の選択は、左端に表示されているキーを入力するか、あるいは↑、↓キーで各項目を選択した後↵を押す

図 5.26　f･1 の入力で表示された「ファイル関係の操作」のメニュー(MENU1)

作成されたテキストをディスクにセーブしてMIFESを終了するには、このメニューの「現テキストのセーブと編集終了」の「E」を選択します。メニューの左端に表示されている文字をキー入力することにより、それぞれのメニュー項目を選択することができます。あるいは ↓ ↑ キーで各項目を直接選択して↵することによっても、目的の項目を選択できます。もし、作業のすべてをキャンセルしてMIFESを終了し、もとの状態のままMS-DOSのプロンプトにもどるには、「現テキストの強制的な編集終了」を選択します。

では E を入力します。作成したテキストは、MIFESの起動時のコマンドラインで指定した「WXYZ.MIF」というファイル名でセーブされ、MIFESを終了してMS-DOSのプロンプトにもどります。ではファイルが作成されたかどうか確認してみましょう。

```
A>DIR WXYZ.MIF            ……作成したファイルを確認する

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

WXYZ     MIF       137  89-09-13    0:32
        1 個のファイルがあります.
   395264 バイトが使用可能です.

A>TYPE WXYZ.MIF           ……作成したファイルの内容を確認する
Good Morning MIFES
おはようマイフェス君
Good afternoon MIFES
こんにちはマイフェス君            入力したテキストによるファイルが作成されている!
Good night MIFES
おやすみＭＩＦＥＳ　また明日

A>
```

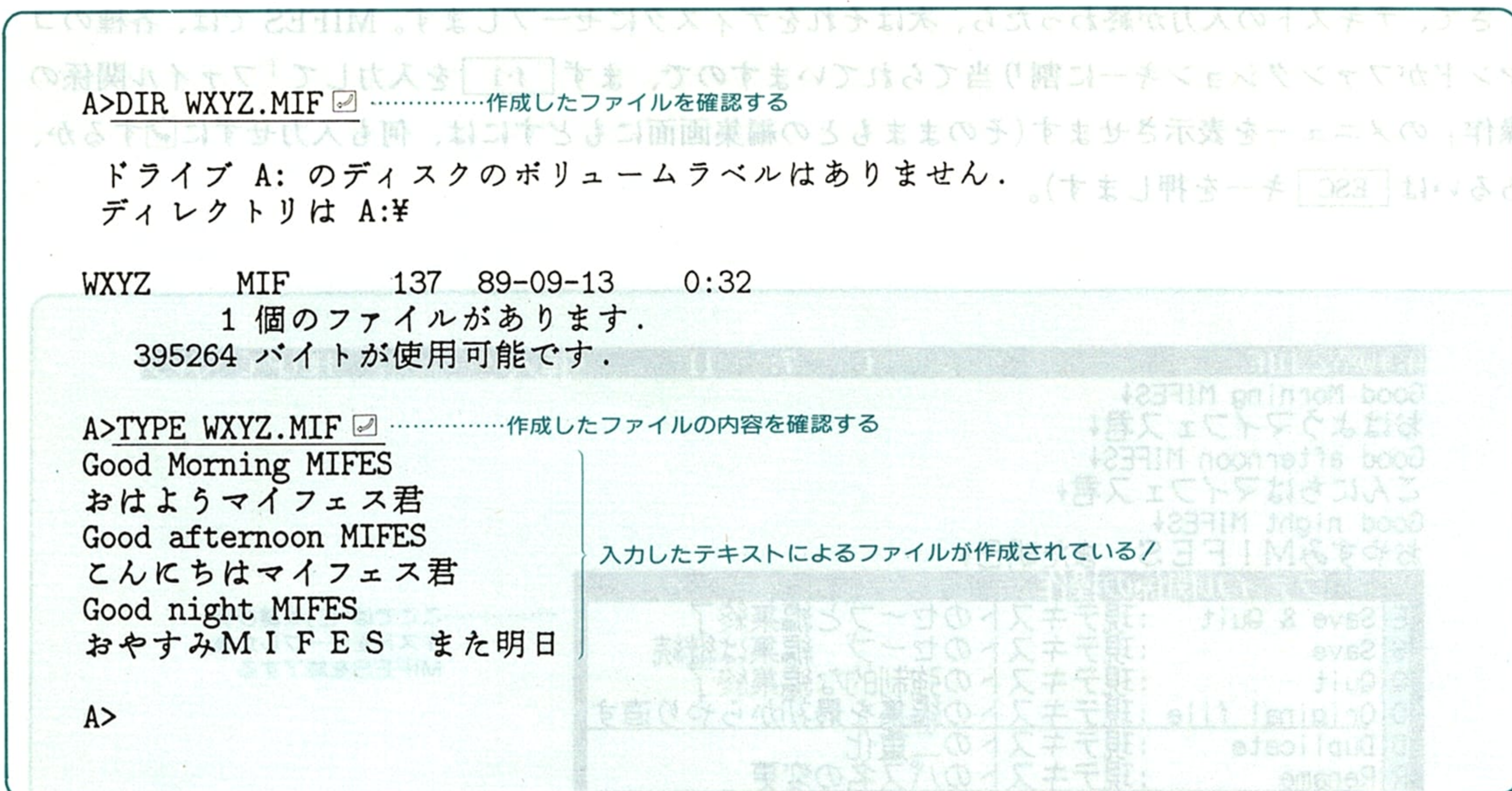



図 5.27　MIFES の終了(Save & Quit)後、作成したファイルの内容を確認する

■ 既存ファイルの編集(追加／変更／削除／読み込み)

ここでは前項で作成した既存ファイル、「WXYZ.MIF」の編集を行ってみましょう。既存ファイルを編集するには、2つの方法で MIFES を起動することができます。1つは MIFES を起動する際のコマンドラインに目的のファイル名を指定する方法です —— 前述の「A>x：MIFES　y：ファイル名」。この場合には、MIFES の起動と同時に、指定されたファイルを読み込んで画面上に表示します。

もう1つの方法は、MIFES を起動するコマンドラインに、目的のファイル名を指定せず、MIFES だけを先に起動する方法です。ここでは、この方法で行ってみましょう。MIFES を次のように起動します。

```
A>MIFES
```

編集対象のファイル名を指定せずに MIFES を起動すると、次の図のように、カレントドライブのカレントディレクトリ上のファイルの一覧表が表示されます。この一覧表から、編集したいファイルを反転カーソルで選択します。目的のファイルがほかのドライブやディレクトリに存在する場合は、画面の「PATH＝xxxx」の指定を、目的のファイルのパスに変更します(次ページの図のコメントを参照)。

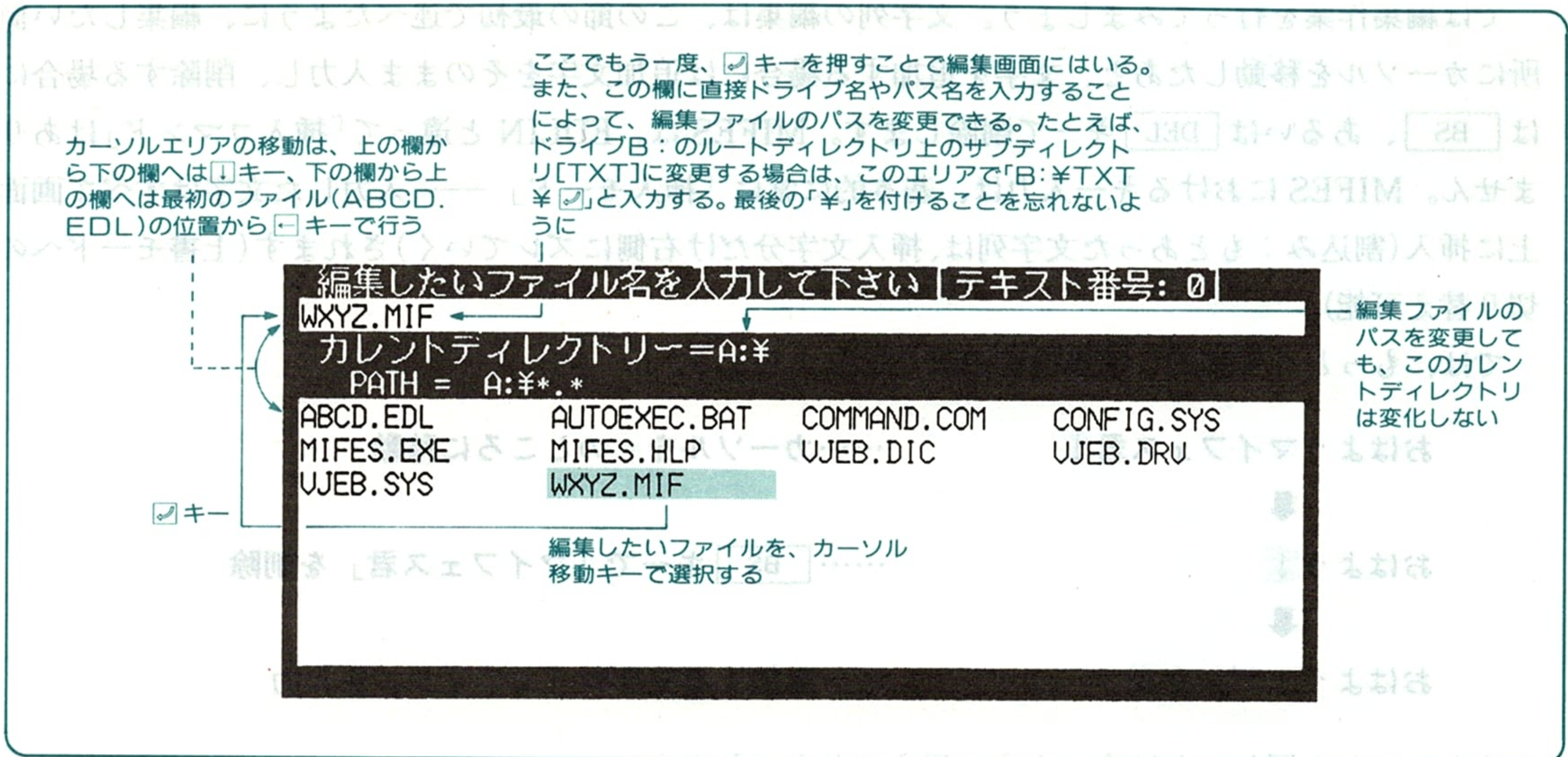



図 5.28　表示されたファイルの一覧表から、編集するファイルを選択する

カーソル移動キーでファイル名を選択し、⏎キーを2回押すことによって、指定したファイルが読み込まれます。

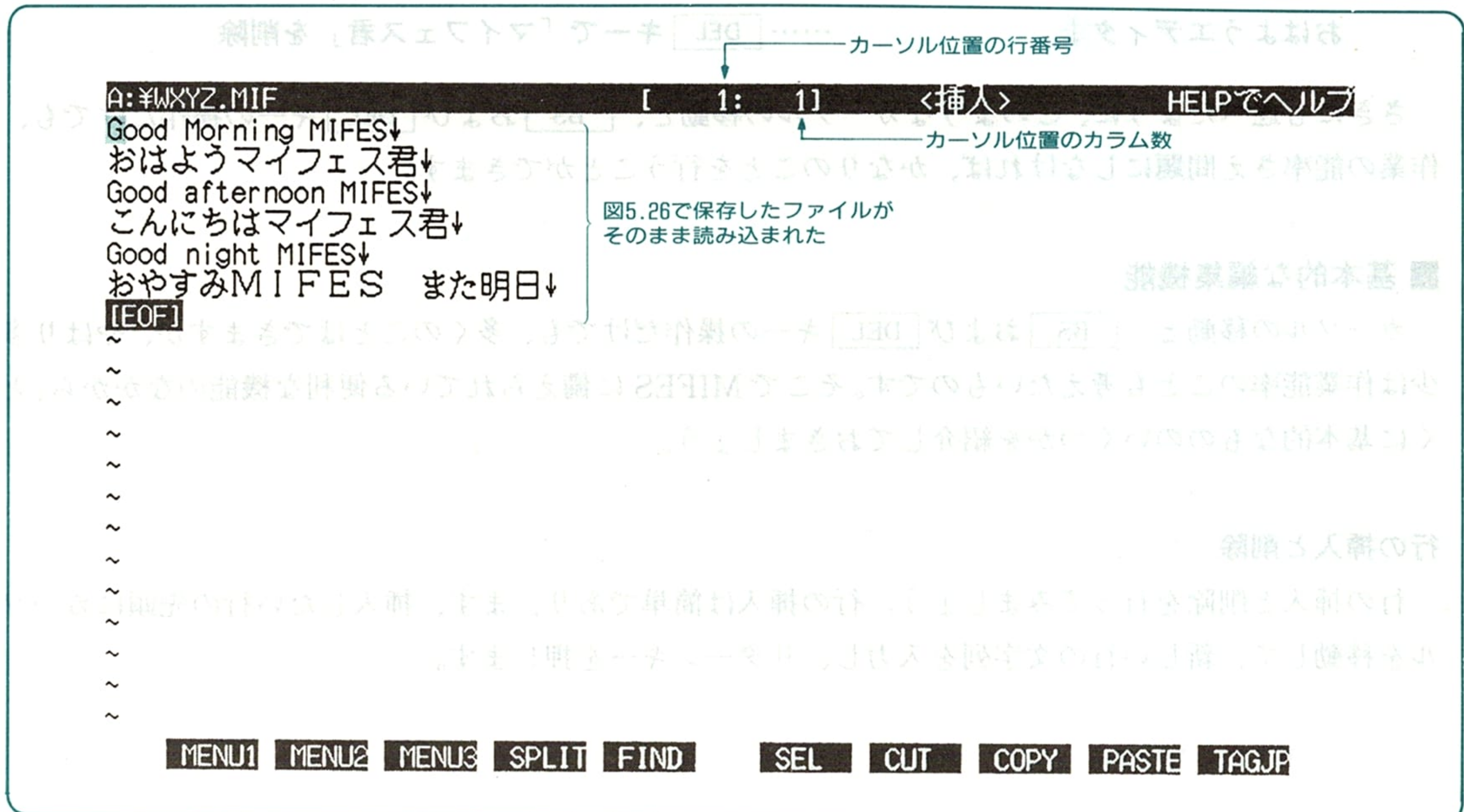



図 5.29　一覧表で指定したファイルが編集画面に読み込まれた

では編集作業を行ってみましょう。文字列の編集は、この節の最初で述べたように、編集したい箇所にカーソルを移動したあと、文字を追加する場合には追加文字をそのまま入力し、削除する場合には［BS］、あるいは［DEL］キーで削除します。MIFESは、EDLINと違って「挿入コマンド」はありません。MIFESにおけるキー入力は、基本的に常に「挿入モード」——入力した文字はすべて画面上に挿入(割込み：もとあった文字列は、挿入文字分だけ右側にズレていく)されます(上書モードへの切り替え可能)。

では、もっとも基本的な編集操作の例を示しましょう。

おはようマイフェス君↓　　……**カーソルを　のところに移動**

⬇

おはよう↓　　……**［BS］キーで「マイフェス君」を削除**

⬇

おはようエディタ↓　　……**追加する文字列「エディタ」を入力**

また、これと同じことは次のように行うこともできます。

おはようマイフェス君↓　　……**カーソルを　のところに移動**

⬇

おはようエディタマイフェス君↓　　……**「エディタ」を入力。挿入される**

⬇

おはようエディタ↓　　……**［DEL］キーで「マイフェス君」を削除**

さきにも述べたように、このようなカーソルの移動と、［BS］および［DEL］キーの操作だけでも、作業の能率さえ問題にしなければ、かなりのことを行うことができます。

■ 基本的な編集機能

カーソルの移動と、［BS］および［DEL］キーの操作だけでも、多くのことはできますが、やはり多少は作業能率のことも考えたいものです。そこでMIFESに備えられている便利な機能のなかから、とくに基本的なもののいくつかを紹介しておきましょう。

行の挿入と削除

行の挿入と削除を行ってみましょう。行の挿入は簡単であり、まず、挿入したい行の先頭にカーソルを移動して、新しい行の文字列を入力し、リターンキーを押します。

おはようエディタ↓　　　　　　　……カーソルを行の先頭に移動

⬇

How are you?おはようエディタ↓　……文字列を入力し、リターンキーを押す

⬇

How are you?↓　　　　　　　　……結果として新しい行が挿入された
おはようエディタ↓

この方法は簡単ですが、行の先頭にカーソルをいちいち移動させなければなりません。そこでカーソルが行内のどこにあっても、CTRL＋N（CTRLキーを押しながらNキーを押す）を入力することにより、カーソル行（カーソルが存在する行）の上に新しい行を挿入することができます。

Good afternoon　E DLIN ↓　　　……ここでCTRL＋Nを押す

⬇

↓　　　　……さきのカーソル行の上に新しい行が挿入され、カーソルが左端に移動する
Good afternoon EDLIN

このようにMIFESのコマンドは、CTRL＋文字キーや、ファンクションキーによって実行されます。それ以外の通常のキーの入力文字は、すべてカーソル位置に挿入されていくわけです。

図 5.30　MIFESのコマンド操作は、ファンクションキーとコントロールキーで

次に、行単位の削除は CTRL ＋ Y で行います。削除したい行にカーソルを移動し（行内のどこでもよい）、CTRL ＋ Y を入力することにより、カーソル行の全体が削除されます。

おやすみMIFES　また明日↓　……削除する行にカーソルを移動する

⬇

[EOF]　……CTRL ＋ Y の入力で行が削除され、次の行にカーソルが移動する

復活機能

復活機能（いわゆる UnDo：アンドゥ）とは、一度削除した文字列を、もとにもどす（復活させる）機能です（もとの位置でなく、任意の位置にもどすこともできる）。間違って削除してしまった場合の修復や、この機能を積極的に利用して、文字列を移動させる場合（いわゆる Move）にも使うことができます。

復活機能を実際に使ってみましょう。削除された文字の復活は、CTRL ＋ P で行います。

こんにちはマイフェス君↓　……「こんにちは」を DEL キーで削除する

⬇

マイフェス君↓　……行末にカーソルを移動する（意識的に行末に復活する）

⬇

マイフェス君は↓　……CTRL ＋ P を入力すると、削除した文字が終わりから順に復活する

⬇

マイフィス君ちは↓

⬇

マイフェス君にちは↓

⬇

マイフェス君んにちは↓

⬇

マイフェス君こんにちは↓

このように、BS キーや DEL キーで削除した文字列は、CTRL ＋ P で復活させることができます。この例では、カーソルを別の位置に移してから復活を行ったので、結果的には「文字列の移動」ということになりました。カーソルがもとの位置でこれを行えば、誤って削除した文字列をもとにもどす、本来の「復活機能」になるわけです。

また復活機能は、行単位の復活にも使うことができます。

Good night M IFES ↓　　　……CTRL + Y で行の削除を行う

⬇

マイフェス君こんにちは↓……1 つ上の行にカーソルを移動する(意識的に別の行に復活する)

⬇

Good night MIFES ↓　　　……CTRL + L で削除した行が復活する
マイフェス君こんにちは↓

このように、CTRL + Y で削除した行は、CTRL + L でカーソルのある行の上に復活させることができます。なお、これらの復活機能は、それぞれ直前に削除したもののみに対して有効です。

検索／置換機能

検索や置換機能は、エディタには必要不可欠の重要な機能です。前節の EDLIN にも備わっていましたが、MIFES の場合は、操作がよりわかりやすくなっています。これらの機能は SHIFT + f･3 (SHIFT キーを押しながら f･3 キーを押す)で行います。まずカーソルをテキストの先頭に移動してから(後述の理由から)、SHIFT + f･3 を押してください。次の画面が表示されます。

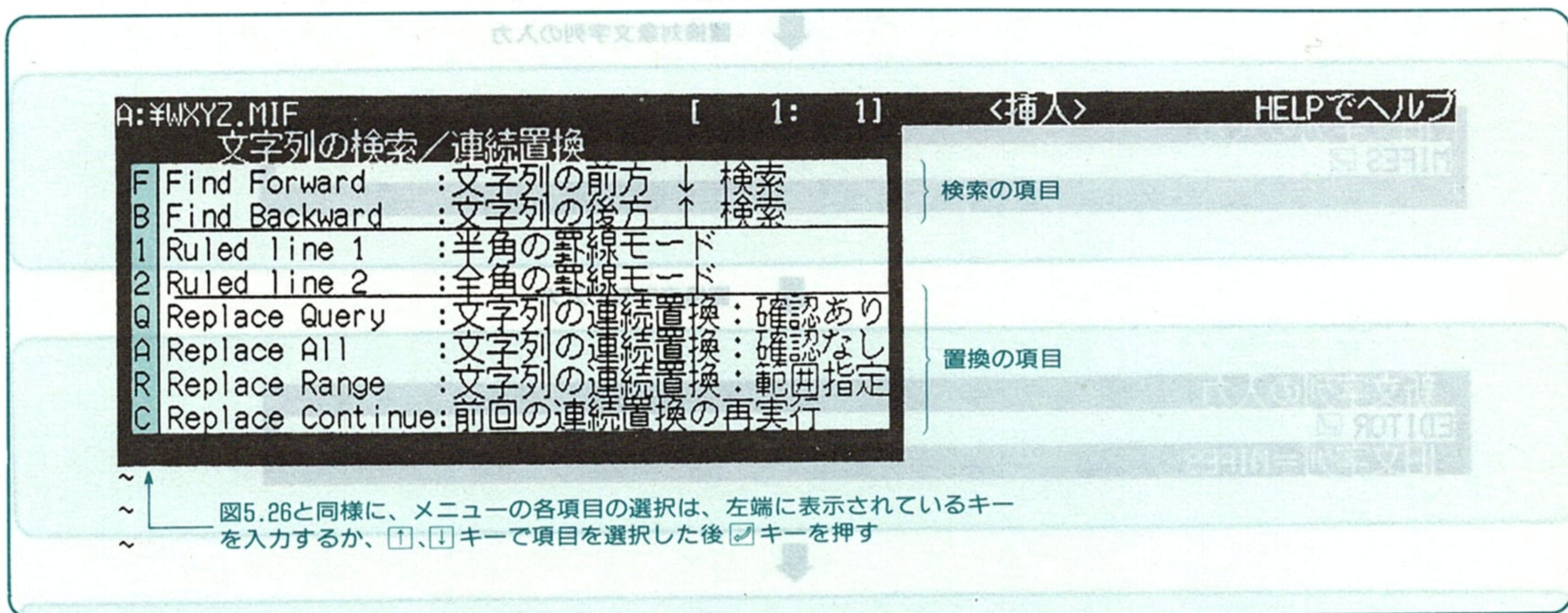
```
A:¥WXYZ.MIF                 [    1:    1]    <挿入>        HELPでヘルプ
      文字列の検索／連続置換
 F Find Forward     :文字列の前方 ↓ 検索
 B Find Backward    :文字列の後方 ↑ 検索
 1 Ruled line 1     :半角の罫線モード
 2 Ruled line 2     :全角の罫線モード
 Q Replace Query    :文字列の連続置換：確認あり
 A Replace All      :文字列の連続置換：確認なし
 R Replace Range    :文字列の連続置換：範囲指定
 C Replace Continue:前回の連続置換の再実行
~
~
~
```


図 5.31　SHIFT + f･3 の入力による検索／置換機能のメニュー

ファイルの保存のときと似たようなメニューが表示されます(そのままもとの編集画面にもどすには、何も入力せずに↵するか、あるいは ESC キーを押します)。このメニューのなかから、左端の文字を入力するか、あるいは ↓ ↑ キーで選択して↵することにより、目的の機能を選択します。ここでは Q を入力して「文字列の連続置換：確認あり」を選択し、実行してみましょう。その実行例を次ページの図 5.32 に示します。

```
A:¥WXYZ.MIF                  [    8:    1]     <挿入>          HELPでヘルプ
Good Morning MIFES↓
おはようエディタ↓
How are you?↓
↓
Good afternoon MIFES↓
Good night MIFES↓
マイフェス君こんにちは↓
[EOF]
```

テキストの先頭にカーソルを移動しておく

```
      文字列の検索／連続置換
F Find Forward      :文字列の前方 ↓ 検索
B Find Backward     :文字列の後方 ↑ 検索
1 Ruled line 1      :半角の罫線モード
2 Ruled line 2      :全角の罫線モード
Q Replace Query     :文字列の連続置換：確認あり
A Replace All       :文字列の連続置換：確認なし
R Replace Range     :文字列の連続置換：範囲指定
C Replace Continue:前回の連続置換の再実行
```

確認ありの連続置換を選択する

置換対象文字列の入力

```
 旧文字列の入力
MIFES
```

置換文字列の入力

```
 新文字列の入力
EDITOR
 旧文字列＝MIFES
```

```
A:¥WXYZ.MIF          Retun key:置換／Space bar:次へ／ESC:置換終了
Good Morning MIFES↓
おはようエディタ↓
How are you?↓
↓
Good afternoon MIFES↓
Good night MIFES↓
マイフェス君こんにちは↓
[EOF]
```

置換対象として検索された文字列が反転表示

- 置換をするときはリターンキー
- 置換をしないときはスペースバー
- 置換作業を終了するときは ESC キー

置換対象文字列が次々に検出されるので、リターンキーもしくはスペースバーで置換するかどうかを選択する

```
A:¥WXYZ.MIF                      3 個の文字列を置換しました  HELPでヘルプ
Good Morning EDITOR↓
おはようエディタ↓
How are you?↓
↓
Good afternoon EDITOR↓
Good night EDITOR↓
マイフェス君こんにちは↓
[EOF]
```

いくつ置換が行われたかを示すメッセージが表示される

すべて置換された／

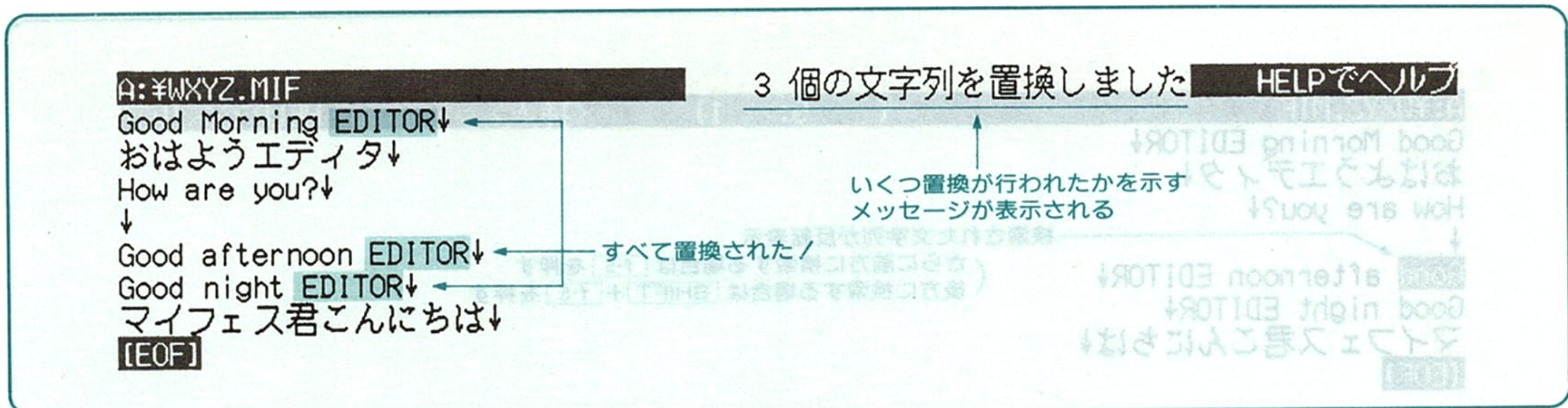



図 5.32　置換の実行例

置換機能は、カーソル位置以降の文字列を置換対象としますので、すべてのテキストを対象としたい場合には、あらかじめカーソルをテキストの先頭に移動しておきます。

次に検索機能を実行してみましょう。同じ SHIFT ＋ f･3 によるメニューで、F または B を選択します。

```
      文字列の検索／連続置換
F Find Forward     :文字列の前方 ↓ 検索
B Find Backward    :文字列の後方 ↑ 検索
1 Ruled line 1     :半角の罫線モード
2 Ruled line 2     :全角の罫線モード
Q Replace Query    :文字列の連続置換：確認あり
A Replace All      :文字列の連続置換：確認なし
R Replace Range    :文字列の連続置換：範囲指定
C Replace Continue:前回の連続置換の再実行
```

あらかじめテキストの先頭にカーソルを移動して、文字列の前方検索を選択する

↓ 検索文字列の入力

```
前方 ↓ 検索する文字列の入力
Good ⏎
```

↓

A:¥WXYZ.MIF　[5: 1]　<挿入>　HELPでヘルプ
Good Morning EDITOR↓
おはようエディタ↓
How are you?↓
↓
Good afternoon EDITOR↓
Good night EDITOR↓
マイフェス君こんにちは↓
[EOF]

検索された文字列が反転表示
さらに前方に検索する場合は f·5 を押す
後方に検索する場合は SHIFT ＋ f·5 を押す

f·5 キーで次々と該当文字列を検索していく

A:¥WXYZ.MIF　文字列が見つかりません　HELPでヘルプ
Good Morning EDITOR↓
おはようエディタ↓
How are you?↓
↓
Good afternoon EDITOR↓
Good night EDITOR↓
マイフェス君こんにちは↓
[EOF]

これ以上、検索文字列が見つからない
場合に表示されるメッセージ

図 5.33　検索の実行例

おなじ文字列をさらに検索したい場合は、f·5（カーソルより後方(逆方向)に検索したい場合は SHIFT ＋ f·5）を入力します。この入力を続けることにより、同じ文字列を次々に検索することが可能です。

編集したテキストのセーブ

ではここで、編集したテキストをディスクにセーブ(保存)してみましょう。ディスクへのセーブは、すでに述べましたが、f·1 を入力してファイル操作関係のメニューを表示させたのち、「現テキストのセーブと編集終了」の E を入力するか、あるいはその項目をカーソル移動キーで選択して⏎します。

ではセーブしたテキストを、DIR コマンドと TYPE コマンドで確認してみましょう。

```
A>DIR WXYZ.*⏎ ……………編集されたファイルを確認する

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

WXYZ     BAK       137  89-09-13    0:32 ……………バックアップファイルが作成されている
WXYZ     MIF       122  89-09-13    1:20 ……………編集されたファイル
        2 個のファイルがあります.
   394240 バイトが使用可能です.

A>TYPE WXYZ.MIF⏎ ……………編集後のファイルの内容を確認する
Good Morning EDITOR
おはようエディタ
How are you?

Good afternoon EDITOR
Good night EDITOR
マイフェス君こんにちは

A>
```

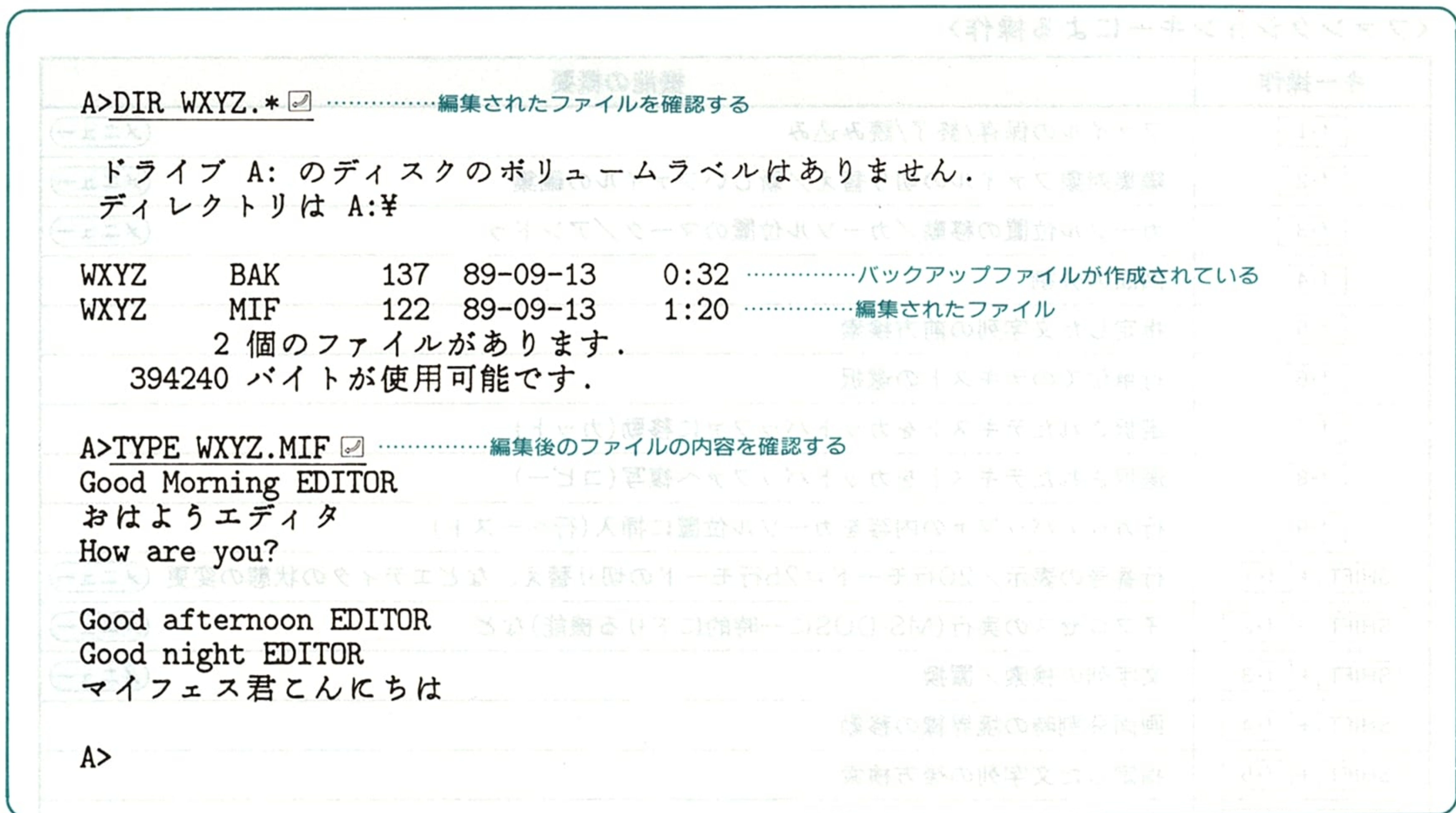



図 5.34 ディスクにセーブしたテキストの確認

このように、もとのファイルはバックアップファイル「.BAK」として残され、編集されたファイルがもとのファイル名でセーブされています。

さて、MIFESの基本的な使い方を一通り紹介しましたが、この程度の知識があれば、かなりの編集作業を行うことができるものと思います。ここで、これまでに登場した機能を含め、MIFESの主要な機能とその操作法の一覧表を次ページに示しておきます。

MIFES(メガソフト社)

〈ファンクションキーによる操作〉

キー操作	機能の概要	
f·1	ファイルの保存/終了/読み込み	メニュー
f·2	編集対象ファイルの切り替え／新しいファイルの編集	メニュー
f·3	カーソル位置の移動／カーソル位置のマーク／アンドゥ	メニュー
f·4	画面の分割	
f·5	指定した文字列の前方検索	
f·6	行単位でのテキストの選択	
f·7	選択されたテキストをカットバッファに移動(カット)	
f·8	選択されたテキストをカットバッファへ複写(コピー)	
f·9	行カットバッファの内容をカーソル位置に挿入(行ペースト)	
SHIFT＋f·1	行番号の表示／20行モード⇄25行モードの切り替え、などエディタの状態の変更	メニュー
SHIFT＋f·2	子プロセスの実行(MS-DOSに一時的に下りる機能)など	メニュー
SHIFT＋f·3	文字列の検索／置換	メニュー
SHIFT＋f·4	画面分割時の境界線の移動	
SHIFT＋f·5	指定した文字列の後方検索	
SHIFT＋f·6	文字単位でのテキストの選択	
SHIFT＋f·7	マクロメニュー1の表示	メニュー
SHIFT＋f·8	マクロメニュー2の表示	メニュー
SHIFT＋f·9	文字カットバッファの内容をカーソル位置に挿入(文字ペースト)	

(メニュー)表示のあるものは各ファンクションキーを押すと機能の一覧が表示され、そのなかから選択を行う

〈CTRL／SHIFT＋文字キーによる操作〉

■カーソルの移動

キー操作	機能の概要
CTRL＋E	カーソルを1行上に移動(↑と同じ)
CTRL＋X	カーソルを1行下に移動(↓と同じ)
CTRL＋D	カーソルを1文字右へ移動(→と同じ)
CTRL＋S	カーソル1文字左へ移動(←と同じ)
CTRL＋C	画面を半画面分(または全画面分)ロールアップ(ROLL UPと同じ)
CTRL＋R	画面を半画面分(または全画面分)ロールダウン(ROLL DOWNと同じ)
SHIFT＋→	カーソルを行の右端に移動
SHIFT＋←	カーソルを行の左端に移動

■行の挿入

キー操作	機能の概要
CTRL＋N	カーソル行の上に1行挿入し、カーソルを左端に移動
SHIFT＋↵	カーソル行の下に1行挿入し、カーソルを左端に移動

■文字列の削除

キー操作	機能の概要
CTRL + H	カーソルの前の1文字を削除（BS と同じ）
CTRL + G	カーソル位置の1文字を削除（DEL と同じ）
CTRL + Y	カーソル位置の行を1行削除 （削除された文字列は削除文字列バッファに移動）
CTRL + K	カーソル位置から行末までの文字列を削除 （削除された文字列は削除文字列バッファに移動）
CTRL + U	行頭からカーソル位置の直前までの文字列を削除 （削除された文字列は削除文字列バッファに移動）

■文字列の復活

キー操作	機能の概要
CTRL + L	削除文字列バッファ（CTRL + Y などで削除された文字列）の内容をカーソル位置に挿入
CTRL + P	削除文字列（BS 、DEL キーなどで削除された文字列）をカーソル位置に1文字ずつ挿入

〈特殊キーによる操作〉

キー操作	機能の概要
INS	挿入モード／上書モードの切り替え
HELP	ヘルプメニューの表示
HOME CLR	2つのファイル間での画面切り替えとカーソル移動
ROLL UP	画面を半画面分（または全画面分）ロールアップ（CTRL + C と同じ）
ROLL DOWN	画面を半画面分（または全画面分）ロールダウン（CTRL + R と同じ）
↑、↓、→、←	カーソルの移動

表 5.2　MIFES の主要な機能とコマンド一覧

■ 複数ファイルの編集とカット&ペースト

MIFES の基本的な機能については、一通り解説しましたが、さらに使いこなしたいという人のために、少し高度な機能 ──「複数ファイルの編集とカット&ペースト」について、参考までに紹介しておきましょう。

MIFES は、複数（最大 10 個まで）のファイルを一度に読み込んで、画面を切り替えながら編集することができます。この機能を使うことにより、あるファイルから別のファイルに文字列を移動したり、コピーしたりすることが簡単に行えます。

では、EDLIN の節で作ったファイル「ABCD.EDL」と、さきほどのファイル「WXYZ.MIF」の 2 つのファイルを同時に編集画面上に読み込んで、簡単な編集操作をいくつか行ってみましょう。2 つのファイルを同時に編集画面上に読み込むには、次の 3 つの方法があります。

① MIFESの起動時に2つのファイルを指定する

A＞MIFES　ABCD.EDL　WXYZ.MIF

この方法は、ファイル名が明確にわかっている場合に便利です。

②ファイル名の指定なしでMIFESを起動し、表示されるファイル選択メニューで2つのファイルを選択する

A＞MIFES

③ 2つのファイルを選択した後、さらに↵キーを押すと、2つのファイルを読み込んで編集画面にはいる

```
編集したいファイル名を入力して下さい　テキスト番号: 0
ABCD.EDL WXYZ.MIF↵
カレントディレクトリー＝A:¥
  PATH =  A:¥*.*
ABCD.EDL     AUTOEXEC.BAT  COMMAND.COM   CONFIG.SYS
MIFES.EXE    MIFES.HLP     VJEB.DIC      VJEB.DRV
VJEB.SYS     WXYZ.BAK      WXYZ.MIF
```

① ↵キー ② ↵キー

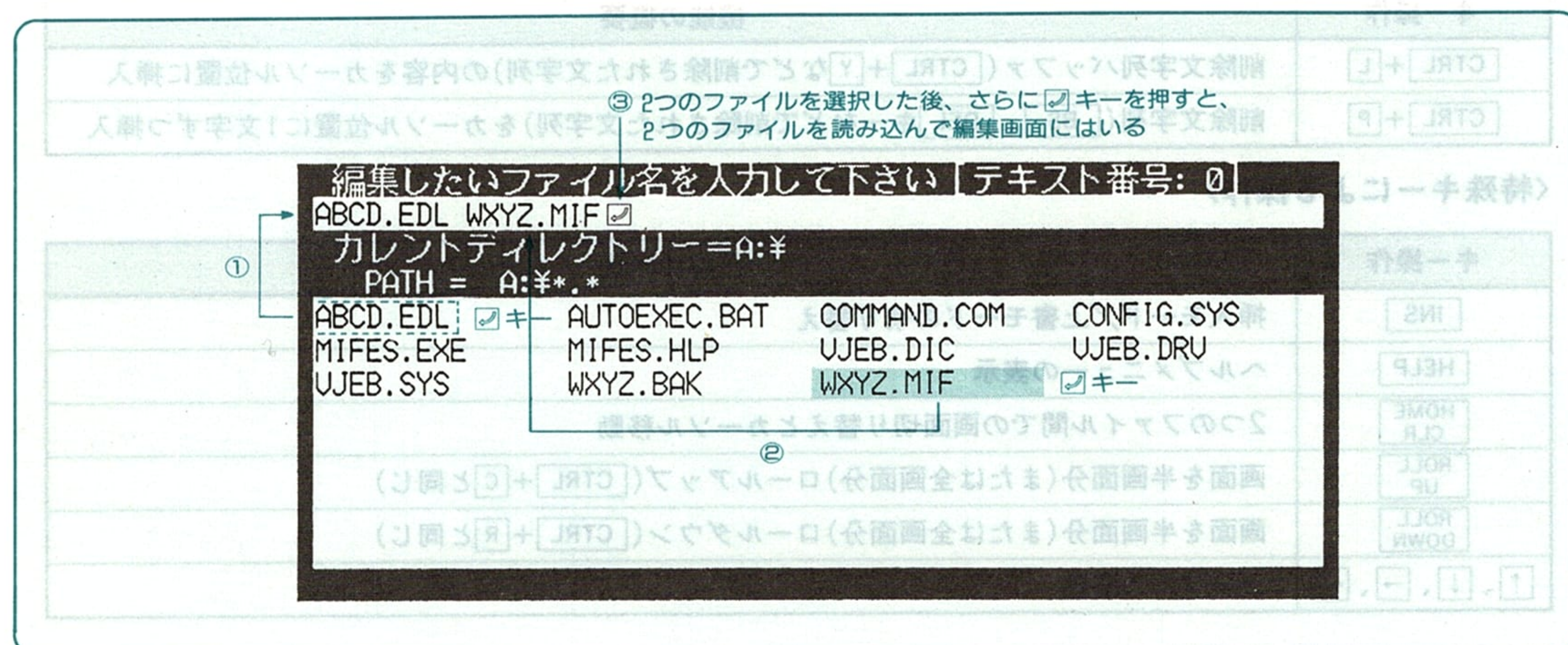



図5.35　ファイル選択メニュー

表示されているファイルのなかから、カーソルで最初のファイルを選択し↵を入力します。続いて2番目のファイルを同様に選択し、↵を2度入力します。

③ MIFESの起動時のコマンドラインで最初のファイルを読み込み、起動したあと f·1 の機能で2番目のファイルを読み込む

A＞MIFES　ABCD.EDL

MIFESが起動したあと、 f·1 で「ファイル関係のメニュー」を選択し、「新しいファイルの編集」を選びます(f·2 の「編集テキストの切り換え」にも同様のメニューがある)。

```
  ファイル関係の操作
E Save & Quit    :現テキストのセーブと編集終了
S Save           :現テキストのセーブ 編集は継続
Q Quit           :現テキストの強制的な編集終了
O Original file  :現テキストの編集を最初からやり直す
D Duplicate      :現テキストの二重化
R Rename         :現テキストのパス名の変更
A All Save & End :全テキストのセーブとエディタ終了
0 End            :強制的なエディタの終了
I Insert file    :カーソル行の前にファイルを読み込む
N New file       :新しいファイルの編集
```

「N」(新しいファイルの編集)を選択

```
 編集したいファイル名を入力して下さい [テキスト番号: 1]
WXYZ.MIF
 カレントディレクトリー=A:¥
```

すでに読み込まれているファイルも
メモリ上の編集エリアに残る

図 5.36 ファイル関係の操作メニュー

この方法は、あるファイルを編集中に別のファイルの内容を見たりコピーしたりするときに利用すると便利です。

以上のいずれかの方法で、2つのファイルを読み込みます。読み込まれた2つのファイルの、どちらを編集対象にするかの切り替えは、[f･2] で行います。[f･2] により次の画面が表示されます。

```
   編集テキストの切り換え(変:変更あり <>:読み込み中)
0  ○ A:¥ABCD.EDL          [ 80:  8:    1]
1  ◎ A:¥WXYZ.MIF          [ 80:  8:    1]
2
3
4
5
6
7
8
9
N New file       :新しいファイルの編集
~
~
~
```

2つのファイルの場合は、
[HOME CLR] キーで切り替えることも可能

◎印が編集対象ファイル

数字を入力することで編集対象ファイルを
瞬時に切り替えることができる

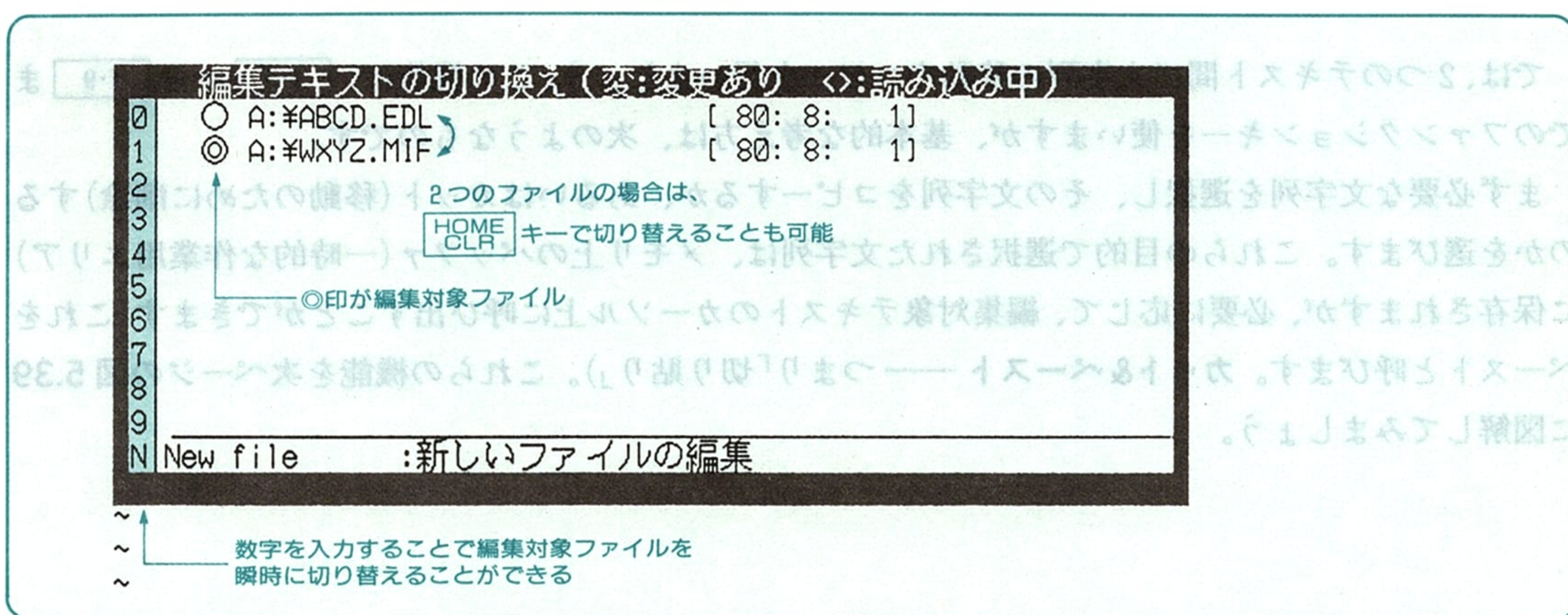



図 5.37 編集テキストの切り替えメニュー

メニューの左端に表示された数字を入力することにより、編集するテキストを瞬時に切り替えることができます。実際にやってみましょう。また、この例のように2つのファイルの場合は、HOME CLR キーを押すことで交互に編集対象テキストを切り替えることができます。

これはカーソル位置の行番号とカラム数

```
A:¥WXYZ.MIF                [   8:   1]   <挿入>        HELPでヘルプ
Good Morning EDITOR↓
おはようエディタ↓
How are you?↓
↓
Good afternoon EDITOR↓
Good night EDITOR↓
マイフェス君こんにちは↓
[EOF]
```

f·2 + 0 ↓　　HOME CLR ↕　　f·2 + 1 ↑

```
A:¥ABCD.EDL                [   8:   2]   <挿入>        HELPでヘルプ
Good morning EDLIN↓
こんにちはエドリン君↓
Insert line↓
挿入行です↓
Good night EDLIN↓
こんにちはエドリン君↓
Insert line↓
おやすみエドリン　また明日↓
[EOF]
```

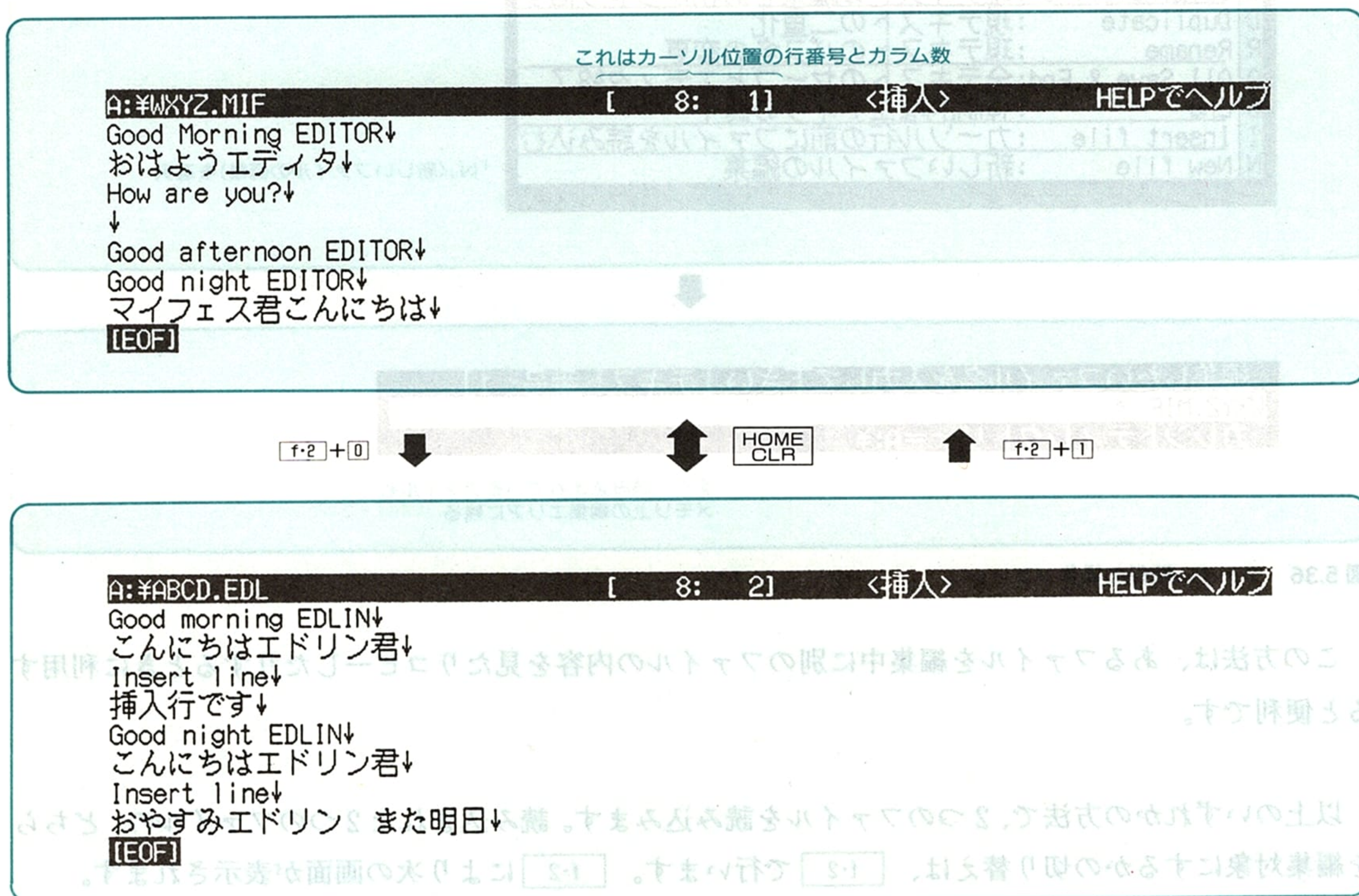



図 5.38　編集テキストの切り替え

では、2つのテキスト間で文字列の移動やコピーを行いましょう。この機能は、f·6 から f·9 までのファンクションキーを使いますが、基本的な考え方は、次のようなものです。

まず必要な文字列を選択し、その文字列をコピーするか、あるいはカット（移動のために削除）するのかを選びます。これらの目的で選択された文字列は、メモリ上のバッファ（一時的な作業用エリア）に保存されますが、必要に応じて、編集対象テキストのカーソル上に呼び出すことができます（これをペーストと呼びます。**カット&ペースト** ――つまり「切り貼り」）。これらの機能を次ページの**図 5.39** に図解してみましょう。

SHIFT + f・6 文字単位の選択

おはよう
こんにちは
おやすみ

f・7 カット

おはよう
こんは
おやすみ

f・8 コピー

おはよう
こんにちは
おやすみ

にち

文字単位のカットやコピーの内容がストアされる

コンピュータ本体

メモリ上のバッファエリア

SHIFT + f・9
文字単位のペースト

おはよう
こんはにち
おやすみ

SHIFT + f・9
文字単位のペースト

おはよう
こんにちは
おやすみ
にち

任意の位置にカーソルを移動することでどこにでも何度でもペーストが可能

f・6 行単位の選択

おはよう
こんにちは
おやすみ

f・7 カット

おはよう

f・8 コピー

おはよう
こんにちは
おやすみ

こんにちは
おやすみ

行単位のカットやコピーの内容がストアされる

コンピュータ本体

メモリ上のバッファエリア

次回のカットやコピーが行われるまで、バッファ上に保存される

この時点で編集対象ファイルを切り替えることにより、任意のファイル間で文字列の移動やコピーが行える

f・9 行単位のペースト

こんにちは
おやすみ
おはよう

f・9 行単位のペースト

おはよう
こんにちは
おやすみ
こんにちは
おやすみ

任意の位置にカーソルを移動することでどこにでも何度でもペーストが可能

図 5.39　カット&ペーストの概念図

では実際に、文字列の移動やコピーを行ってみます。2つのテキスト間の移動は、前述の HOME CLR キーで行います。

ファイルWXYZ.MIF

```
A:¥WXYZ.MIF
Good Morning EDITOR↓
おはようエディタ↓
How are you?↓
↓
Good afternoon EDITOR↓
Good night EDITOR↓
マイフェス君こんにちは↓
[EOF]
```

ファイルABCD.EDL

```
A:¥ABCD.EDL
Good morning EDLIN↓
こんにちはエドリン君↓
Insert line↓
挿入行です↓
Good night EDLIN↓
こんにちはエドリン君↓
Insert line↓
おやすみＥＤＬＩＮ　また明日↓
[EOF]
```

文字単位でコピー&ペースト
行単位でカット&ペースト
文字単位でコピー&ペースト
HOME CLR キーによるテキストの切り替え

（作業結果）

```
A:¥WXYZ.MIF
Good Morning EDITOR↓
おはようエディタ↓
How are you?↓
Insert line↓
挿入行です↓
Good afternoon EDITOR↓
Good night EDITOR↓
マイフェス君こんにちは↓
[EOF]
```

（作業結果）

```
A:¥ABCD.EDL
Good morning EDLIN↓
こんにちはエドリン君エディタ↓
挿入行です↓
Good night EDLIN↓
こんにちはエドリン君エディタ↓
Insert line↓
おやすみＥＤＬＩＮ　また明日↓
[EOF]
```

図 5.40　2つのファイル間で文字列の移動とコピーを行う

カットやコピーした文字列は、メモリ上のバッファに存在していますから、何度でも同じ文字列を呼び出すことができます。複数の箇所に同じ文字列をコピーするときなどに利用すると便利です。

また MIFES は、2つのテキストファイルを画面上に同時に表示することができます。これは、画面を上下2つに分割することにより（これを「ウィンドウ」と呼ぶ）、それぞれのテキストを表示します。1つのテキストの表示領域は画面の半分になりますが、2つのテキストを同時に確認しながら文字列の移動やコピーができるので便利です。

ではこの操作を実際に行ってみましょう。次ページの図 5.41 のように、 f·4 を入力することにより画面が分割されます。

```
         ┌─(反転していない)
A:¥WXYZ.MIF                    [   9:   1]     <挿入>          HELPでヘルプ
Good Morning EDITOR↓
おはようエディタ↓
How are you?↓
Insert line↓
挿入行です↓
Good afternoon EDITOR↓
Good night EDITOR↓                        HOME CLR キーで編集対象ファイルを切り替える
マイフェス君こんにちは↓
[EOF]
~
         ┌─現在の編集対象ファイル（反転文字）
~
A:¥ABCD.EDL                    [   8:   1]     <挿入>          HELPでヘルプ
Good morning EDLIN↓
こんにちはエドリン君エディタ↓
挿入行です↓
Good night EDLIN↓
こんにちはエドリン君エディタ↓
Insert line↓
おやすみＥＤＬＩＮ　また明日↓
[EOF]
~
~
                       ┌─ f･4 キーで画面分割のオン／オフを行う
~
 MENU1 MENU2 MENU3 SPLIT FIND   SEL  CUT  COPY  PASTE  TAGJP
```

図 5.41　画面を分割して２つのテキストを同時に表示する

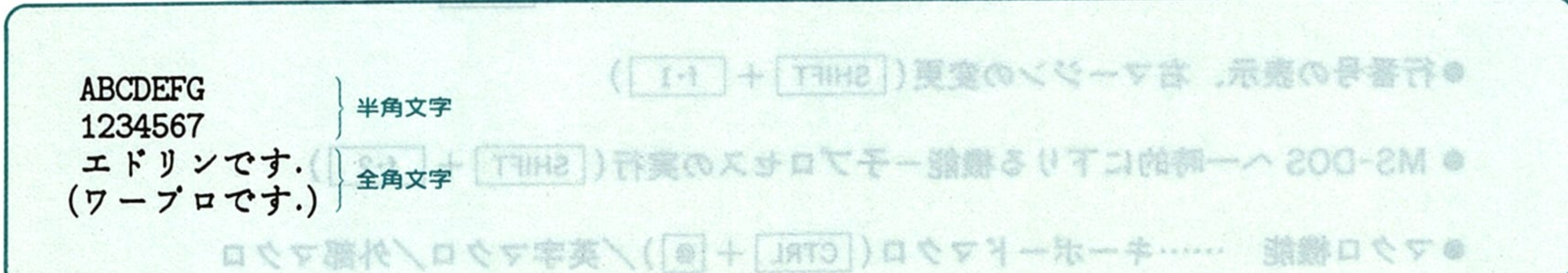
２つのウィンドウ間のカーソルの移動は、前述と同様に HOME CLR で行います。ここで紹介したカット&ペーストの機能は、２つのテキスト間だけでなく、１つのテキストのなかでの文字列の移動やコピーにも使えます（長いファイルには便利です）。前述の復活機能とともに、必要に応じて使いわけるとよいでしょう。

参考までに、本項で登場したもの以外に、MIFES でよく使われる機能を羅列しておきます。

- カーソルのジャンプ　……テキストの最初と最後の行へジャンプ（ f･3 のメニュー）
マークした行へのジャンプ（ f･3 のメニュー）
指定した行番号へのジャンプ（ ESC ＋数字キー）
- 行番号の表示、右マージンの変更（ SHIFT ＋ f･1 ）
- MS-DOS へ一時的に下りる機能－子プロセスの実行（ SHIFT ＋ f･2 ）
- マクロ機能　……キーボードマクロ（ CTRL ＋ @ ）／英字マクロ／外部マクロ
- ヘルプ機能（ HELP ）

5.4 ワードプロセッサによるテキストファイルの作成

日本語ワープロや英文ワープロのほとんどは、エディタとしても使うことができます。「ワープロもエディタも同じようなことをするものだからあたりまえではないか」と思われるかもしれません。しかしワープロには、その文書ファイルのデータ形式が、各ワープロソフト独自の形式をとるものも多く、ディスプレイに表示される文書は同じでも、文書データの内部が異なっているため、そのままでは「テキストファイル」(純粋な文字ファイル)として使えないものがあります。しかし、現在のたいていのワープロには、作成したそれぞれの文書ファイルを、テキスト形式のファイルとして出力したり、各ワープロ独自の形式の文書ファイルを、テキスト形式に変換したりする機能が備えられています。

ここで、エディタによって作成されるテキスト形式のファイルと、ワープロによって作成される文書ファイルとの、内部データの違いを簡単にまとめておきましょう。

- **エディタによって作成されるテキストファイル**
  ファイルの内部データが、純粋な文字コード(復帰/改行コードなどのアスキーコードに準拠した制御コードを含む)からできている。

- **ワープロによって作成される文章ファイル**
  ファイルの内部データが、文字コードのほかに、それぞれのワープロ独自の形式のヘッダー部や、文字飾り、罫線などをはじめとする各種の識別/制御コードを含んでいる。ただし、一太郎のバージョン3以前のものなど、ヘッダー部や各種の制御部などは別ファイルとし、メインファイルはテキスト形式をとるものもある(バージョン4では独自の形式の1本のファイルに統合された)。

では簡単な文章を、エディタの「EDLIN」と、日本語ワープロの「新松」、および「一太郎バージョン3」、「バージョン4」を使って作成し、それぞれのファイルの内部データの違いを見てみましょう。次に示す簡単な内容の文書ファイルを作成します。

```
ABCDEFG            } 半角文字
1234567            }
エドリンです.       } 全角文字
(ワープロです.)     }
```

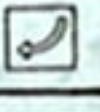



図 5.42　サンプル用の文書ファイル

この文書ファイルが、それぞれ「TEST.EDL」(エディタによるもの)、「TEST.BUN」(新松によるもの)、「TEST.JXW」(一太郎バージョン3によるもの)、「TEST.JSW」(一太郎バージョン4によるもの)の各ファイル名で、ディスク上に作成されています。このそれぞれのファイルの内部を、MS-DOSの「ファイルDUMPプログラム」(DUMP.EXEあるいはDUMP.COM)を使って調べてみましょう。

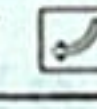
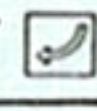
```
A>DIR TEST.*          ……各ファイルの確認

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

TEST     EDL       35  89-08-21  19:13 ……ラインエディタ「EDLIN」によるテキストファイル
TEST     BUN      316  89-08-21  19:18 ……ワードプロセッサ「新松」による文書ファイル
TEST     JXW       35  89-08-21  19:23 ……ワードプロセッサ「一太郎Ver.3」による文書ファイル
TEST     JSW     8192  89-08-21  19:39 ……ワードプロセッサ「一太郎Ver.4」による文書ファイル
        4 個のファイルがあります.
   543744 バイトが使用可能です.

A>DUMP TEST.EDL       ……ラインエディタ「EDLIN」で作成されたテキストファイルの
                        内部データを、DUMPプログラムで見る(ダンプする)
                                    コード
 Dump Version 3.00                 復帰 改行
             A  B  C  D  E  F  G  ↓  ↓  1  2  3  4  5  6  7
00000000    41 42 43 44 45 46 47 0D-0A 31 32 33 34 35 36 37    ABCDEFG..1234567
00000010    0D 0A 83 47 83 68 83 8A-83 93 82 C5 82 B7 81 44    ..エドリンです.
00000020    0D 0A 1A エ    ド    リ    ン    で    す    。    ...
 コード                     (全角文字はシフトJISコード)

A>DUMP TEST.BUN       ……ワードプロセッサ「新松」で作成された文書ファイルの内部をダンプする

 Dump Version 3.00
              CTRL+Z
             ↓                         A  B  C  D  E  F  G  1  2
00000000    1A 4D 02 01 15 13 00 41-42 43 44 45 46 47 31 32    .M.....ABCDEFG12
00000010    33 34 35 36 37 83 8F 81-5B 83 76 83 8D 82 C5 82    34567ワープロです
            3  4  5  6  7
00000020    B7 81 44 00 00 00 00 00-00 00 00 00 00 00 D7 D8    . ...........ﾗﾘ
00000030    78 00 00 00 02 01 00 00-04 01 00 00 06 01 00 00    文章以外の各種の
00000040    08 01 00 00 0A 01 00 00-01 00 01 00 02 00 2C 01    データ部が付加さ
00000050    2C 01 C8 00 C8 00 00 0F-27 00 28 00 64 00 64 00    れている
     (ただし、新松にはテキスト形式のファイルを直接出力する機能あり。図5.45を参照)
   ∫

A>DUMP TEST.JXW       ……ワードプロセッサ「一太郎Ver.3」で作成された文書ファイルの内部をダンプする

 Dump Version 3.00
             A  B  C  D  E  F  G  コード 1  2  3  4  5  6  7
00000000    41 42 43 44 45 46 47 0D-0A 31 32 33 34 35 36 37    ABCDEFG..1234567
00000010    0D 0A 83 8F 81 5B 83 76-83 8D 82 C5 82 B7 81 44    ..ワープロです.
00000020    0D 0A 1A ワ    ー    プ    ロ    で    す    。    ...
 コード      EDLINで作成したファイルと同一形式のテキストファイルである(文章以外の
             各種のデータ部は「.ATR」と「.CTL」という拡張子を持つ別のファイルに収められている)
```

```
A>DUMP TEST.JSW ↵         ……ワードプロセッサ「一太郎Ver.4」で作成された
                              文書ファイルの内部をダンプする

 Dump Version 3.00
                    ファイルが暗号化されているのでダンプしても、内容を知ることはできない
00000000  44 4F 43 00 00 00 00 00-20 20 20 20 20 20 20 20   DOC.....
00000010  00 00 01 00 01 89 09 12-00 00 00 00 00 00 00 02   ................
00000020  00 00 00 00 00 00 00 00-01 00 03 04 89 09 12 01
                                  ∫
00001DF0  C0 1E 80 3D 00 7B 00 F6-8A 3F 14 7F 28 FE DA 2F   ﾀ..=.{...?..(.ﾚ/
00001E00  B4 5F 68 BF 5A AD 3E 89-F6 C1 66 50 CC A0 12 92   ｴ_hｿZｭ>怪ﾁfPｱ..聴
00001E10  AE F7 D6 3C AC 79 58 F3-3A 35 74 6A E8 D4 5A 7A    ..<ｬyX.:5tj斅Zz
                                  ∫
00001FD0  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
00001FE0  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
00001FF0  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
       (ただし、一太郎Ver.4にはテキスト形式のファイルを直接出力する機能あり)
A>
```

図 5.43a　エディタおよびワープロによって作成された文書ファイルの内部データの違い

次に、これらのファイルをTYPEコマンドでタイプアウトしてみましょう。テキスト形式のファイルであれば正常にタイプアウトされます。

```
A>TYPE TEST.EDL ↵  ……EDLINによるファイルをTYPEコマンドでタイプアウトしてみる
ABCDEFG
1234567            } テキスト形式なので正常にタイプアウトされる
エドリンです.

A>TYPE TEST.BUN ↵  ……新松によるファイルをタイプアウトしてみる
                   ……ファイルの先頭に CTRL + Z が挿入されているので、何も
                       表示されずにプロンプトにもどる
A>TYPE TEST.JXW ↵  ……一太郎Ver.3によるファイルをタイプアウトしてみる
ABCDEFG
1234567            } テキスト形式なので正常にタイプアウトされる
ワープロです.

A>TYPE TEST.JSW ↵  ……一太郎Ver.4によるファイルをタイプアウトしてみる
DOC                9@     99"JXW.JEx                              "99    99
99       99      #       9@      99      S      9@      9      Z      9@     9@

                 文章以外のデータが付加され、ファイルが暗号化されて
                 いるために正常にタイプアウトできない

                         ■ｧ           ]ｿrｫ+帙忙r鑽倶Wｺ芋ｳ06dAuｵ■■ﾆ&敷:9B
          ﾆ&敷妝g■zｱ■~ｿ遮n■ ･.
A>
```

図 5.43b　それぞれの文書ファイルをTYPEコマンドでタイプアウトする

ワープロによる文章ファイルには、文字コード以外に各種のコードが含まれているものがあることがわかりました。このような形式のファイルは、自動スタート・バッチファイルや、CONFIG.SYS ファイル、それにアセンブラや各種言語のソースファイルなどに使うことはできません。

そこで、このようなワープロには、たいてい、編集済みファイルをテキスト形式でディスクに出力(保存・セーブ)する機能や、「ワープロ文書ファイル⇄テキストファイル」の形式変換機能などが付属しています。**図 5.44** はこのことを示したものです。

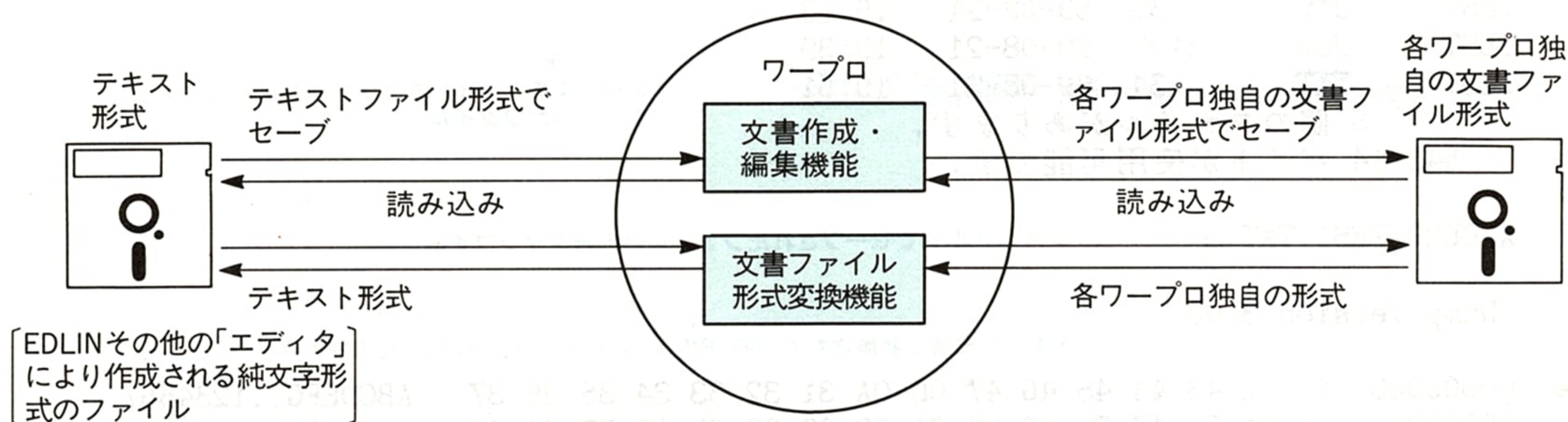



図 5.44　ワープロ文書ファイル⇄テキストファイルの形式変換の考え方

では一例として、さきほどの新松によるワープロの文章ファイルを、テキスト形式に変換してみましょう。新松(松 86 も同じ)の場合は形式変換プログラムを必要とせず、目的の文書ファイルを、いったん編集画面に読み込み、保存形式に「テキスト」を選択して保存終了すればよいのです(つまり、テキストファイル形式でディスクにセーブする。一太郎バージョン 4 の場合も同様)。

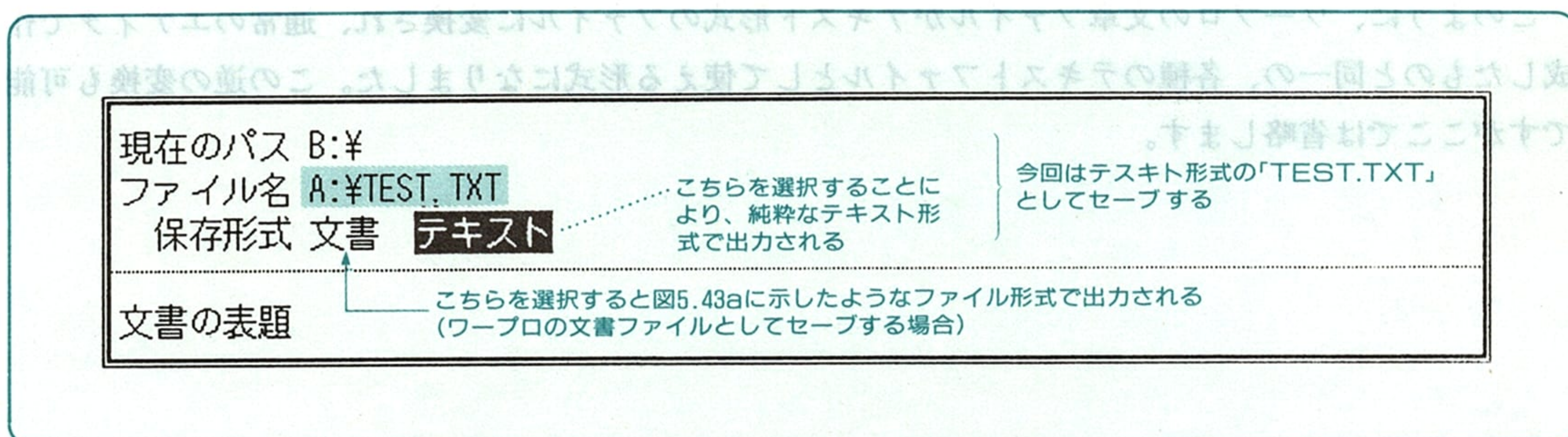



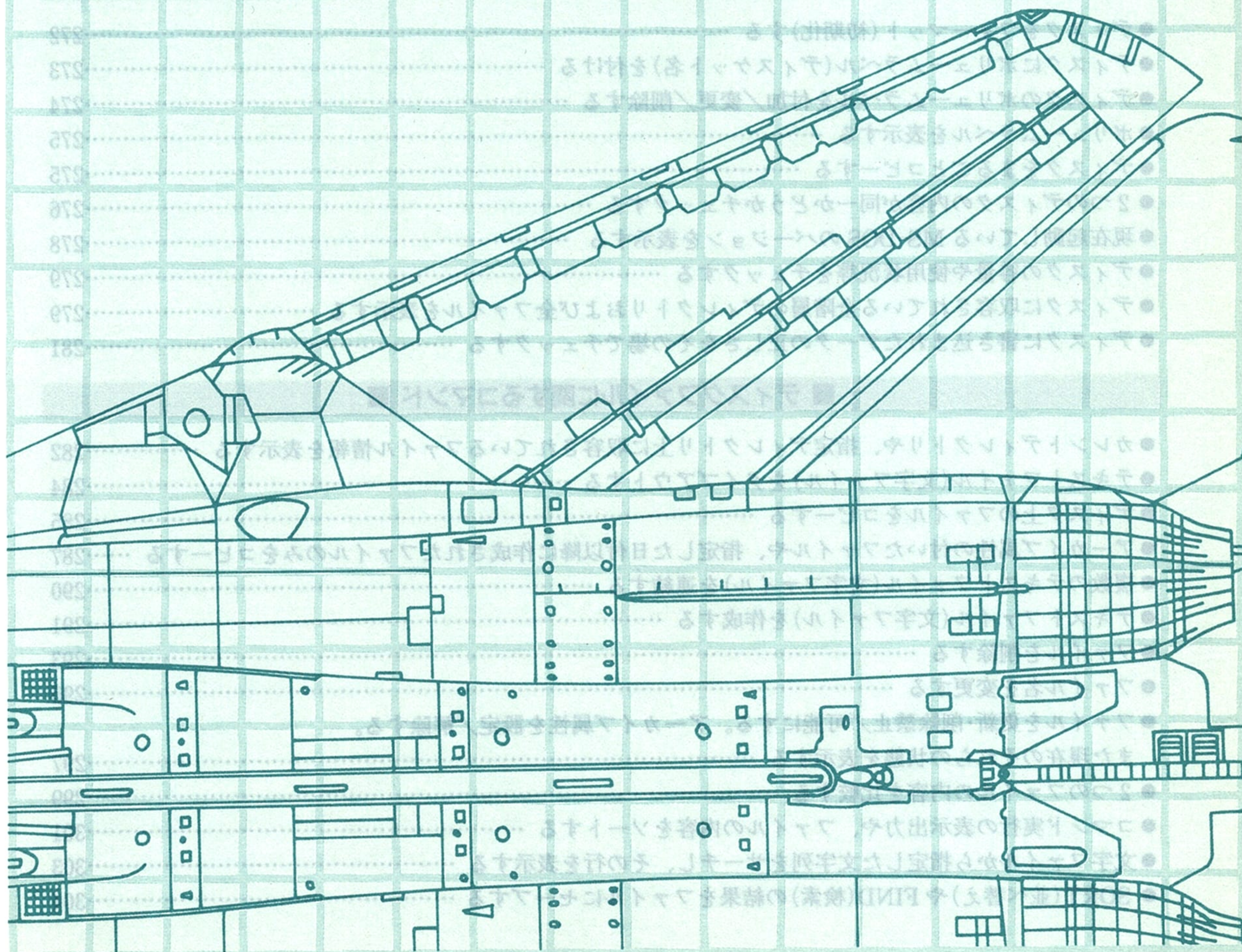
図 5.45　ワープロの文書ファイルをテキスト形式のファイルでセーブする

ではテキスト形式でセーブされたファイル「TEST.TXT」を確認してみましょう。

```
A>DIR TEST.*↵ ……作成されたファイルを確認する
 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

TEST     EDL        35  89-08-21   19:13
TEST     BUN       316  89-08-21   19:18
TEST     JXW        35  89-08-21   19:23
TEST     JSW      8192  89-08-21   19:39
TEST     TXT        34  89-08-21   19:51 ……新松の文書ファイルを、テキスト形式でセーブしたファイル
         5 個のファイルがあります.
    543714 バイトが使用可能です.

A>DUMP TEST.TXT↵ ……テキスト形式でセーブされたファイルの内容をダンプする

 Dump Version 3.00
                テキスト形式に変換されている。図5.43aのEDLINで作成した内容と同一
00000000  41 42 43 44 45 46 47 0D-0A 31 32 33 34 35 36 37   ABCDEFG..1234567
00000010  0D 0A 83 47 83 68 83 8A-83 93 82 C5 82 B7 81 44   ..エドリンです.
00000020  0D 0A                                              ..

A>TYPE TEST.TXT↵ ……TYPEコマンドでタイプアウトする
ABCDEFG
1234567        テキスト形式なので正常にタイプアウトされる
ワープロです.

A>
```

図 5.46　テキスト形式でセーブされたワープロの文書ファイルの内部データを確認する

このように、ワープロの文章ファイルがテキスト形式のファイルに変換され、通常のエディタで作成したものと同一の、各種のテキストファイルとして使える形式になりました。この逆の変換も可能ですがここでは省略します。

6章
MS-DOS主要コマンド目的別解説

本章では、MS-DOS バージョン 3.3 に基づいた主要コマンドを、リファレンスとしてそのまま利用できる形式で解説します。ここでは、各コマンドの紹介を、コマンド名の ABC 順ではなく、行う作業や仕事による種類別にまとめ、それぞれのグループから目的のコマンドの機能や使い方を知るという**逆引き方式**で解説しています。

ここで登場するコマンドは、ビジネスソフトなどを始めとする一般的なアプリケーションプログラムの運用上、何かにつけて必要性があると思われる主要コマンドであり、あまり一般的でないコマンドや、ソフトウェア開発者が専門的に扱うレベルのコマンドなどは除いてあります。

◇ 一般コマンド目的別リファレンス ◇

■ ディスク全体に関するコマンド ■

■ ディスクファイルに関するコマンド ■

■ プリントアウトに関するコマンド ■

■ RS-232C インターフェイスの入出力に関するコマンド ■

■ 階層ディレクトリに関するコマンド ■

■ スクリーン表示関係 ■

■ MS-DOS システムおよびコンピュータシステム関係 ■

■ バッチ処理関係 ■

■本章の読み方

本章は、MS-DOSに関する初歩的な知識は、すでにあることを前提にしています。ここで示す実行例は、基本的にはドライブA：にMS-DOSの標準的なシステムディスク(日常使う常用ディスクで、日本語入力システムはオリジナルとは別のものに入れ替えてある)をセットし、ドライブB：には、「入門MS-DOS」の11章で作成した階層ディレクトリをもつディスク(実習するには何でもよいが)を必要に応じてセットして行ったものです(コマンドによっては、ディスクを入れ替えて実行する場合もある)。そのディスクの階層構造の内容を下の図に示します。

また、ここでリファレンスとして示すコマンド形式の代表例は、カレントドライブをA：として、各種の外部コマンド(外部プログラムやユーザープログラム)はドライブA：のカレントディレクトリ上のものを実行する場合を想定しています。これらの外部プログラムが、そのほかのドライブやディレクトリ上にある場合は、そのドライブ名やディレクトリ名(パス名)を、外部プログラム名の前に付けなければなりませんが、そのような場合のコマンドの書式例は、記述が繁雑になるだけなので一部のものを除き省略してあります。

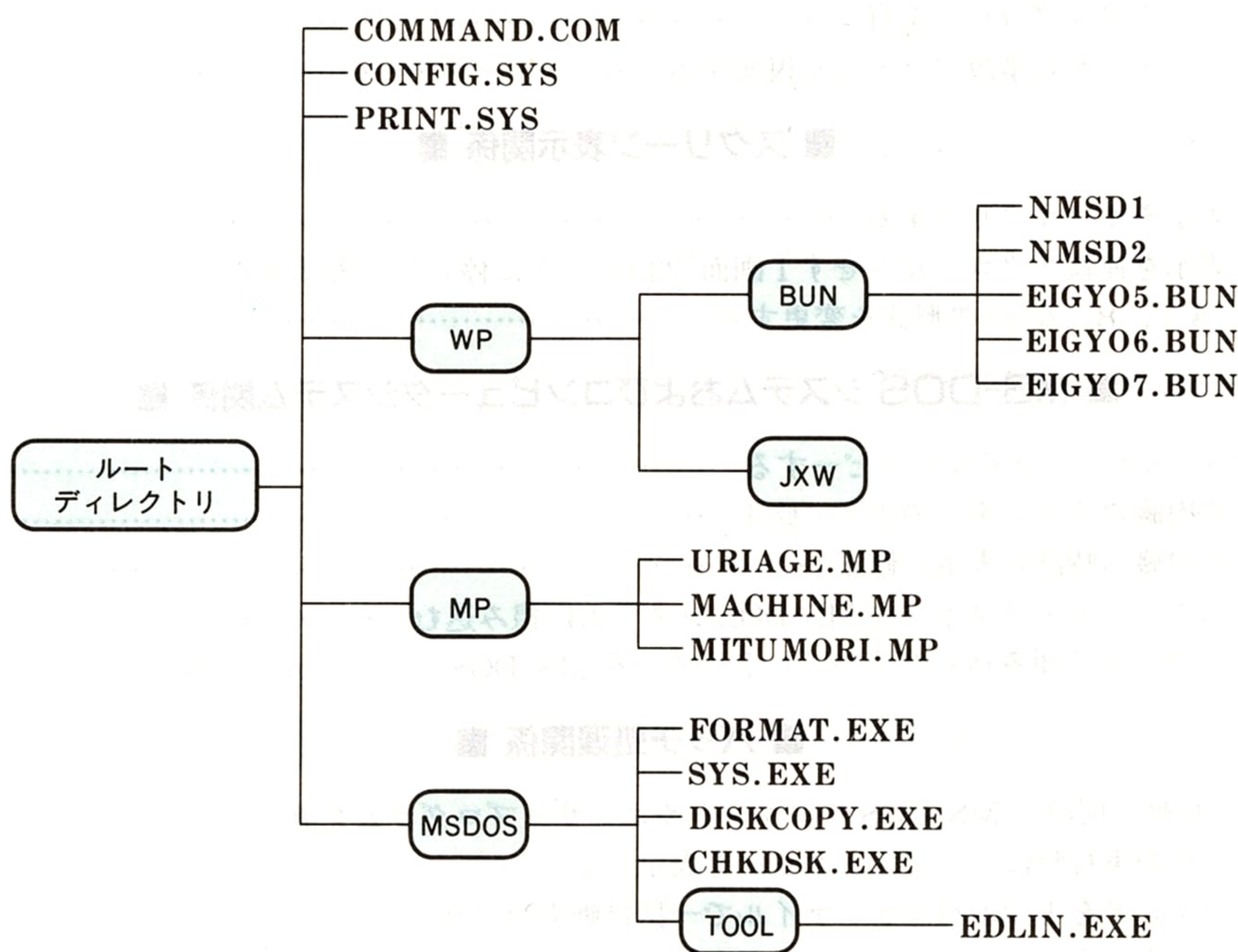



図6.1　本章の階層ディレクトリ関係の実行例で使用するディスクの内容

次に、ここでのコマンドリファレンスで使われる用語や記号について説明しておきましょう。

＜3.*x*＞ ……………………………各種のコマンドで、MS-DOS のバージョン 3.*x* 以降に備えられたものを示す

A＞ ………………………………………MS-DOS のプロンプトで、代表例としてドライブ A：がカレントドライブということで話を進めている

***x*：、*y*：** ………………………………「A：」「B：」などの、ドライブ名を示す。これを省略した場合は、カレントドライブが指定される。また逆に、これらのドライブ名は、それがカレントドライブである場合は記述を省略できる

パス名¥ファイル名 ………………「パス名」(¥MSDOS¥TOOL など)で指定されるディレクトリ上のファイル「ファイル名」(ABCD.XYZ など)を示す。「パス名」を省略した場合は、カレントディレクトリ上の「ファイル名」が指定される。また逆に、これらの「パス名」は、それがカレントディレクトリである場合は記述を省略できる

パス名¥ワイルドカード ………上記の「ファイル名」にワイルドカード(*.COM、DATA??.*など)を使ったもの。ただし、ワイルドカードを使ったコマンド例は原則として省略してあり、ワイルドカードの使用が可能なものは、各コマンドの項目の最初に、そのただし書きがある

パス名…………………………………「¥MSDOS¥TOOL」などの、目的のディレクトリやファイルに到達するまでの階層ディレクトリの"経路"。ここではドライブ名とファイル名を含めていない。この「パス名」を省略した場合は、カレントディレクトリが指定される

_ ……………………………………………スペースの入力を示す(入力しなくても実行可能なものもあるが)

⏎……………………………………………キャリッジリターンの入力を示す

なお、外部コマンドのプログラムファイルのファイルタイプ(拡張子)は、MS-DOS のバージョンや、各機種のシステムディスクにより、「.EXE」と「.COM」の違いがあります。ここでは、PC-9800 シリーズのシステムディスク(バージョン 3.3)を例にしていますが、この違いはまったく気にする必要はありません。また、画面のメッセージなどにも違いがあります。

■ ディスク全体に関するコマンド ■

ディスクをフォーマット(初期化)する

［外部プログラム　FORMAT.EXE］

A＞FORMAT_x：⏎
　ドライブx：上のディスクをフォーマット処理する。

［解説］　購入したディスクや、ほかのシステムで使用していたディスクを、MS-DOS上で使えるように、ディスク全面にMS-DOSのディスク形式によるトラックやセクタ情報を書き込む。このフォーマット処理は、最初に一度だけ行えばよい。ただし、使用済みのディスクを全面クリアするためにも、しばしば行われる。ハードディスクのフォーマット処理は、スイッチ「/H」を付けた「A＞FORMAT_/H⏎」により行われる(161ページの4.2章を参照)。また、FORMATコマンドにスイッチ「/V」や「/S」を付ければ、同時にボリュームラベルを登録したり、MS-DOSシステムをコピーすることも可能(後述)。

```
A>FORMAT B:⏎ ……カレントドライブ(A：)上のFORMATプログラムを起動して、
Format Version 4.10   ドライブB：上のフロッピーディスクをフォーマット処理する

新しいディスクをドライブ B: に挿入し
どれかキーを押してください⏎ ……何を入力してもよい。この時点で CTRL + C を入力すると、
                                当プログラムをキャンセルしてMS-DOSにもどる
ディスクのタイプは  1 : 640(KB)  2 : 1(MB)  =  2⏎ ……2DDのフロッピーなら1、
                                               2HDのフロッピーなら2を入力する
目的のディスクは 1MB FD です

フォーマットが終了しました
0  10  20  30  40  50  60  70  80  90  100
████████████████████████████████████████████(%)

   1250304 バイト  全ディスク容量
   1250304 バイト  使用可能ディスク容量

別のディスクをフォーマットしますか(Y/N) N⏎ ……Yを入力すると続けて作業ができる
A>
```

図6.2　ディスクをフォーマット処理する

ディスクにボリュームラベルを付ける

[外部プログラム　FORMAT.EXE にスイッチ「/V」を付けて実行]

A＞FORMAT_*x*：/V ⏎

ドライブ*x*：上のディスクをフォーマットし、かつボリュームラベルを登録する。

[解説]　個々のフロッピーディスクに、その内容などを識別するための名前「ボリュームラベル」(ディスク名)を付ける。ハードディスクの場合は，「/H」スイッチのなかで指定する。付ける名前の最大文字数は11文字で、日本語入力による漢字の1文字は、2文字に数える。使用可能なキャラクタは、通常のファイル名の場合と同じであるが、ボリュームラベルにはスペースを使うことができる。この機能は、フォーマット処理とともに実行され、単独では実行できない。単独でボリュームラベルを操作するには「LABEL コマンド」を使う(次項参照)。

```
A>FORMAT B: /V⏎
Format Version 4.10

新しいディスクをドライブ B: に挿入し
どれかキーを押してください⏎

ディスクのタイプは　 1 : 640(KB)　2 : 1(MB)　=　2⏎

目的のディスクは 1MB FD です

フォーマットが終了しました
0  10  20  30  40  50  60  70  80  90  100
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■(%)

    1250304 バイト　全ディスク容量
    1250304 バイト　使用可能ディスク容量

ボリュームラベルを入力してください
漢字<全角>は5文字、英数字<半角>は11文字まで、必要なければリターンキー

ボリュームラベル＝電話帳ＤＢ⏎

別のディスクをフォーマットしますか(Y/N) N⏎
A>
```


図6.3　ボリュームラベルを登録する

ディスクのボリュームラベルを付加／変更／削除する

［<3.*x*>外部プログラム　　LABEL.EXE］

A>LABEL␣*x*：ラベル名⏎

ドライブ *x*：上のディスクにボリュームラベル(ディスク名)を付ける。以前のボリュームラベルは新しい「ラベル名」に置き換えられる。

A>LABEL␣*x*：⏎

ボリュームラベルの入力要求にしたがって、ラベル名を入力すれば、それがドライブ *x*：上のディスクに付けられる。また、ラベル名を入力せず、そのまま⏎した場合は、ドライブ *x*：上のディスクの現在のラベル名が削除される。

［解説］　MS-DOS のバージョン 2.*x* では、ボリュームラベルはディスクフォーマット・プログラムの実行時でなければ付加／変更できなかったが、当プログラムにより、フォーマット処理に関係なく、フロッピーディスク、ハードディスクとも、いつでも自由に付加／変更／削除が可能になった。ラベル名の文字数は、半角の場合は 11 文字までであり、漢字の 1 文字は 2 文字に数える。使用可能キャラクタは、通常のファイル名の場合と同じであるが、ボリュームラベルにはスペースを使うことができる。

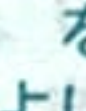
```
A>LABEL B:⏎

ドライブ B: のディスクのボリュームラベルは PC9800 SYS

ディスクのボリュームラベルを入力してください.
漢字<全角>は 5 文字, 英数字<半角>は 11 文字まで
必要なければ <改行キー> : MS-SYS V33⏎

A>DIR B:⏎

 ドライブ B: のディスクのボリュームラベルは MS-SYS V33
 ディレクトリは B:¥

COMMAND  COM    24931  88-07-13    0:00
～～～～～～～～～～～～～～～～～～～～～～～～～～～～
現在のボリュームラベルを削除します.
よろしいですか <Y/N>? Y

A>
```

- A>LABEL B:⏎ ……このように、ドライブ名の後にラベル名を指定しない場合は、ドライブB：上のディスクのボリュームラベルを表示し、必要があれば変更できる
- PC9800 SYS ……ドライブB：には、バージョン3.3のMS-DOSシステムディスクがセットされている
- MS-SYS V33⏎ ……新しくこのようなボリュームラベルに変更する
- A>DIR B:⏎ ……ドライブB：に対してDIRコマンドを実行
- MS-SYS V33 ……ボリュームラベルが変更されている。VOLコマンド「VOL B：⏎」を実行して確認してもよい
- 現在のボリュームラベルを削除します. ……ボリュームラベルを入力せず、⏎のみを入力した場合はこのようなメッセージが表示され、ボリュームラベルの削除や、そのままの状態で当プログラムを終了することができる

図 6.4　ディスクのボリュームラベルを変更する

ボリュームラベルを表示する

［内部コマンド　　VOL］

A＞VOL_*x*：⏎

ドライブ*x*：上のディスクのボリュームラベルを表示する。

［解説］　任意のドライブ上のディスクに付けられたボリュームラベル(ディスク名)を表示する。また、このボリュームラベルはDIRコマンドによっても表示される。

```
A>VOL B:⏎ ……………ドライブB：上のディスクのボリュームラベルを表示する

 ドライブ B: のディスクのボリュームラベルは 電話帳ＤＢ
                                            前実行例(図6.3)で作成したディスク
A>
```

図6.5　ボリュームラベルを表示する

ディスクをまるごとコピーする

［外部プログラム　　DISKCOPY.EXE］

A＞DISKCOPY_*x*：_*y*：⏎

ドライブ*x*：上のディスクの全内容を、そっくりドライブ*y*：上のディスクにコピーする。

［解説］　外形の大きさではなく、トラック数やセクタ数が同じタイプのディスク間(8インチ2D／5インチ2HD／3.5インチ2HDの相互、5インチ2DD／3.5インチ2DDの相互など)で、ディスク上の全データを、フォーマット済みのディスクへ、トラックからトラックへまるごとコピーする。つまり、そっくり同じ状態のディスクができ上がる。コピーに際しては、ほとんどのディスクコピー・プログラムが、コピーされたデータのリード・アフター・ライトによるベリファイ(チェック)を行っている。

```
A>DISKCOPY A: B:↵  ……………カレントドライブ(A：)上のDISKCOPYプログラムを起動し、
                          ドライブA：の全内容をそっくりB：にコピーする

DISKCOPY version 4.00

ディスクのコピーを行います

送り側ディスクをドライブ A: に挿入してください
受け側ディスクをドライブ B: に挿入してください
準備ができたらどれかのキーを押してください x ……………CTRL+Cを入力すると当プログラムを
                                                キャンセルしてMS-DOSにもどる
                                          ↑
                                   この時点で双方のディスクを自由に入れ替え可
照合中      残り   0 トラックです
0  10  20  30  40  50  60  70  80  90  100
███████████████████████████████████████████(%)

コピーは終了しました
もう一度実行しますか(Y/N)N↵

A>
```

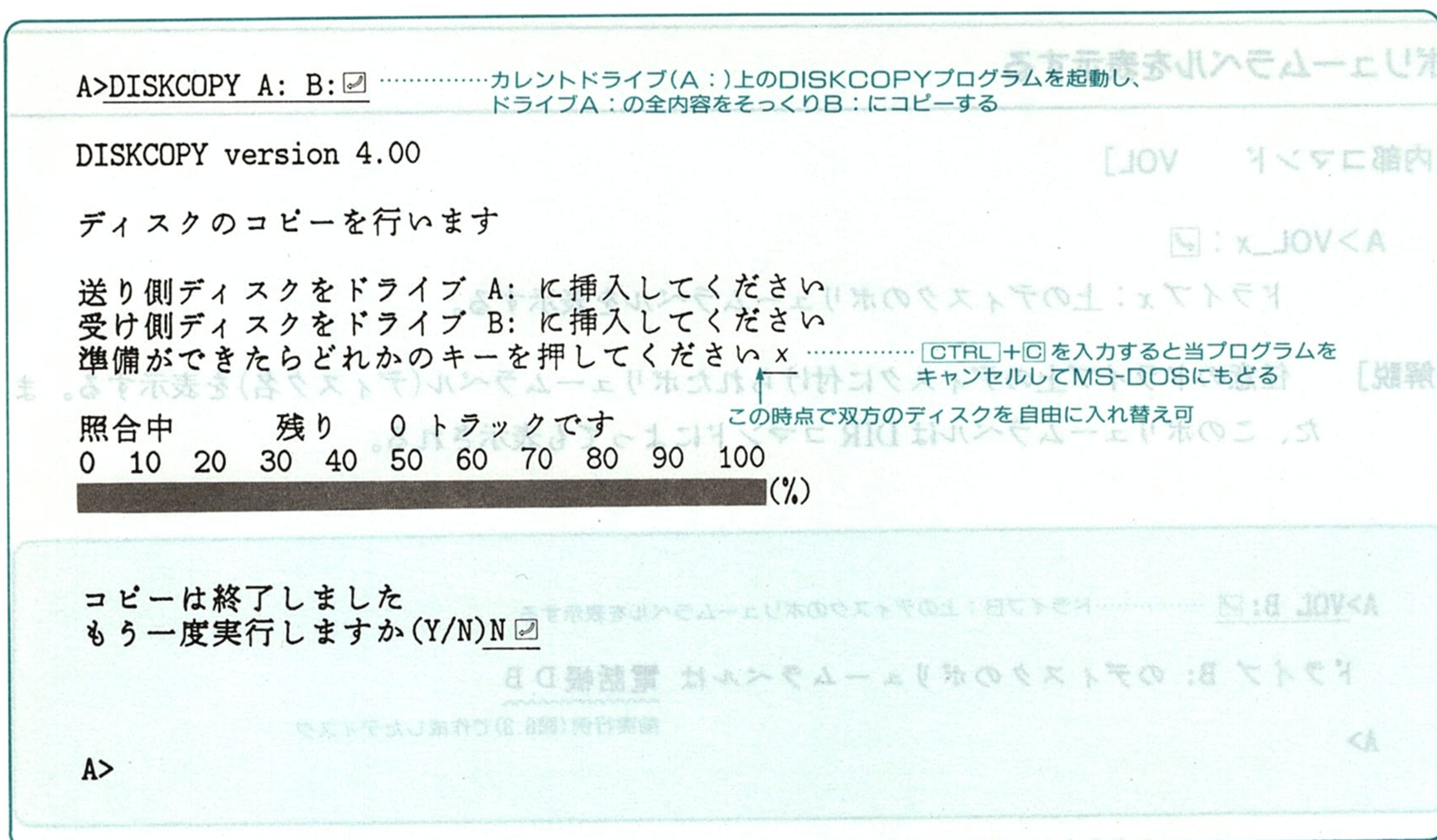



図6.6　ディスクをまるごとコピーする

2つのディスク間の内容が同一かどうかチェックする

［外部プログラム　　DISKCOPY.EXE にスイッチ「/V」を付ける］
　　　　　　　　　（機種によってはこの機能がないものもある）

A＞DISKCOPY_*x*：_*y*：/V↵
　ドライブ*x*：とドライブ*y*：の2つのディスクの内容をトラック単位で比較する。

［解説］　同じ種類の2つのディスク(DISKCOPY コマンドでコピーが可能な種類同士)の内容が、同一かどうかをディスクのトラック対トラックで比較し、その結果を表示する。ファイル単位のチェックではないことに注意。前項の DISKCOPY プログラムでコピーしたままのディスクであれば、改めて当コマンドでチェックする必要はない(コピー時にベリファイされているため)。スイッチ「/V」を付け忘れると、ディスクコピーが実行されてしまうので、実行には十分注意すること。

```
A>DISKCOPY A: B:/V⏎
```
……前実行例（図6.6）にスイッチ「/V」を付けたもの。同一内容のディスクを比較した例。機種によってはこの「/V」の機能がないものもある。不注意で「/V」を忘れて実際にコピーしてしまわないよう十分注意すること！

```
DISKCOPY version 4.00

ディスクの照合を行います

送り側ディスクをドライブ A: に挿入してください
受け側ディスクをドライブ B: に挿入してください
準備ができたらどれかのキーを押してください x
```
（この時点で任意のディスクに入れ替えてよい）

```
照合中　　残り　0 トラックです
0  10  20  30  40  50  60  70  80  90  100
                                           (%)

照合は終了しました
もう一度実行しますか(Y/N)N⏎
```
……エラー表示がない場合は一致している

```
A>DISKCOPY A: B:/V⏎
```
……内容の異なるディスクを比較した例

```
DISKCOPY version 4.00

ディスクの照合を行います

送り側ディスクをドライブ A: に挿入してください
受け側ディスクをドライブ B: に挿入してください
準備ができたらどれかのキーを押してください x

照合中　　残り 154 トラックです
0  10  20  30  40  50  60  70  80  90  100
                                           (%)

トラック　0 の内容が異なっています
中止 <A> ,強行 <I> ? I
```
……「強行」は「続行」の意味

```
照合中　　残り 142 トラックです
0  10  20  30  40  50  60  70  80  90  100
                                           (%)

トラック 12 の内容が異なっています
中止 <A> ,強行 <I> ? A

照合は失敗しました
もう一度実行しますか(Y/N)N⏎
```
……「照合は一致しませんでした」という意味

```
A>
```

図 6.7　2 つのディスク全面の内容を比較する

現在起動している MS-DOS のバージョンを表示する

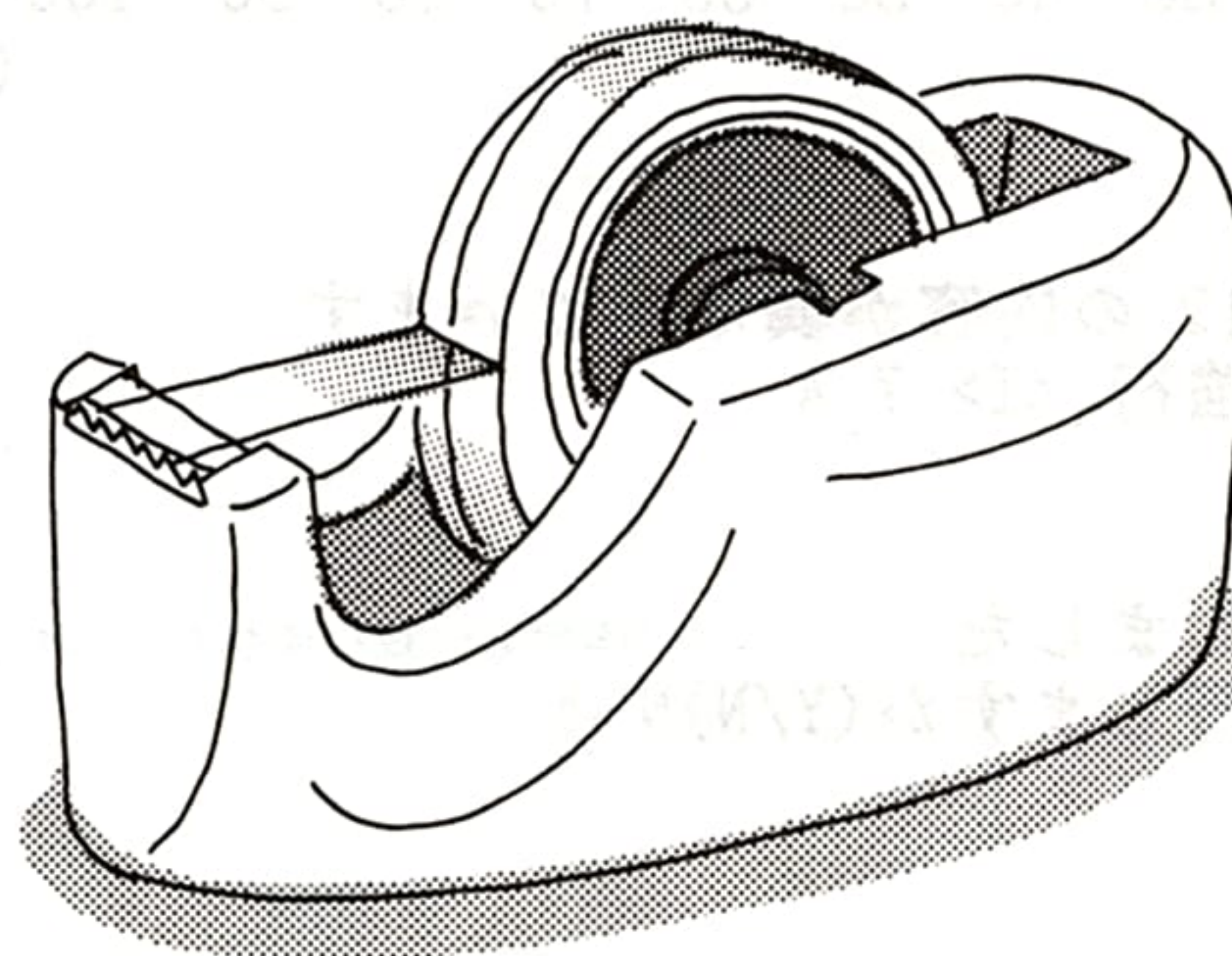
［内部コマンド　　VER］

A＞VER ⏎

現在起動している MS-DOS のバージョン No.を表示する。

［解説］　バージョンの異なる MS-DOS を実行する機会が増えた今日、必要性の高いコマンドになってきた。表示されるのは、現在動作中の MS-DOS のバージョンであり、ディスク上のシステムのバージョンでないことに注意。

```
A>VER ⏎ ……………“現在起動している”MS-DOSのバージョンが表示されるため、ドライブ名などはない
MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30
A>
```

図 6.8　MS-DOS のバージョン No.を表示する

ディスクの容量や使用状況等をチェックする

［外部プログラム　CHKDSK.EXE］

A＞CHKDSK_*x*：⏎

ドライブ*x*：上のディスクの諸元やその使用状況などを表示する。

［解説］　ディスクの諸元や使用状況、それにコンピュータのメインメモリの容量などをチェックする。複数のパーティションに分割されたハードディスクの諸元を調べるために、当コマンドが使われる機会が増えてきた。

```
A>CHKDSK B:⏎ ……………カレントドライブ(A：)上のCHKDSKプログラムを起動し、
                      ドライブB：上のディスクの使用状況などをチェックする

 1250304 バイト ： 全ディスク容量
   95232 バイト ： 2 個のシステムファイル
    6144 バイト ： 6 個のディレクトリ
  399360 バイト ： 16 個のユーザーファイル
  749568 バイト ： 使用可能ディスク容量

  655360 バイト ： 全メモリ ……………コンピュータ内のメインメモリの容量、現在は640Kバイト
  439808 バイト ： 使用可能メモリ ……………同ユーザーエリアの容量

A>
```

図 6.9　メインメモリの容量やディスクの使用状況等を表示する

ディスクに収容されている全階層のディレクトリおよびその全ファイルを表示する

［外部プログラム　CHKDSK.EXE にスイッチ「/V」を付ける］

A＞CHKDSK_*x*：/V⏎

ドライブ*x*：上のディスクに登録されている全ディレクトリ、全ファイルを表示する。

［解説］　DIR コマンドによるファイル情報の表示は、カレントディレクトリ、あるいは指定ディレクトリ上に直接存在するもののみが対象となるが、当コマンドでは、指定ドライブ上の全ディレクトリと、その全ファイル(システムファイルや隠しファイルなども)が対象となる。つまり、ディスク上のすべてのファイルの登録状況を知ることができる。

```
A>CHKDSK B:/V
ディレクトリ B:¥
        B:¥IO.SYS*
        B:¥MSDOS.SYS*
        B:¥COMMAND.COM
        B:¥CONFIG.SYS
        B:¥PRINT.SYS
ディレクトリ B:¥WP
ディレクトリ B:¥WP¥BUN
        B:¥WP¥BUN¥NMSD1
        B:¥WP¥BUN¥NMSD2
        B:¥WP¥BUN¥EIGYO5.BUN
        B:¥WP¥BUN¥EIGYO6.BUN
        B:¥WP¥BUN¥EIGYO7.BUN
ディレクトリ B:¥WP¥JXW
ディレクトリ B:¥MP
        B:¥MP¥URIAGE.MP
        B:¥MP¥MACHINE.MP
        B:¥MP¥MITUMORI.MP
ディレクトリ B:¥MSDOS
        B:¥MSDOS¥FORMAT.EXE
        B:¥MSDOS¥SYS.EXE
        B:¥MSDOS¥DISKCOPY.EXE
        B:¥MSDOS¥CHKDSK.EXE
ディレクトリ B:¥MSDOS¥TOOL
        B:¥MSDOS¥TOOL¥EDLIN.EXE

  1250304 バイト : 全ディスク容量
    95232 バイト : 2 個のシステムファイル
     6144 バイト : 6 個のディレクトリ
   399360 バイト : 16 個のユーザーファイル
   749568 バイト : 使用可能ディスク容量

   655360 バイト : 全メモリ
   433968 バイト : 使用可能メモリ

A>
```


図6.10　ディスクの全ディレクトリ、全ファイルの登録状況を表示する

ディスクに書き込まれたデータの正しさをその場でチェックする

［内部コマンド　VERIFY］

A＞VERIFY_ON ↵

ディスクへの書き込みのすべての場合において、リード・アフター・ライトによるベリファイ(チェック)を行い、書き込まれたデータが正しいことを保証する。

A＞VERIFY_OFF ↵

さきのコマンドの実行によりONとなったベリファイをOFFにする。

A＞VERIFY ↵

現在のベリファイの状態がONであるかOFFであるかを表示する。

［解説］　ディスクへの書き込み時、書き込まれたデータをその場で読み出して、メモリ上のもとのデータと比較し、書き込みエラーがなかったことをチェックする。このチェックをベリファイと呼ぶ。ディスクの信頼性は十分高いので、普通はこれをONにする必要はない(MS-DOSが起動した時点ではOFFになっている)。非常に重要な絶対にエラーがあってはならない場合に使うと安心できる。ただし、書き込み速度は1.5倍ほど遅くなる。

```
A>VERIFY ↵ ……………現在のベリファイ設定状態を表示する
VERIFYは <off> です. ……………MS-DOSが起動した時点ではOFFである

A>VERIFY ON ↵ ……………ONに設定する

A>VERIFY ↵ ……………状態のチェック
VERIFYは <on> です. ……………ONになっている

A>
```

図6.11　ディスクへ書き込まれたデータをその場でチェックするベリファイ機能をON／OFFする

■ ディスクファイルに関するコマンド ■

カレントディレクトリや、指定ディレクトリ上に収容されているファイル情報を表示する

［内部コマンド　DIR　:　ファイル名にワイルドカードの使用可能］

A＞DIR_x：⏎

ドライブx：のカレントディレクトリ上のすべてのファイル情報を表示する。

A＞DIR_x：ファイル名⏎

ドライブx：のカレントディレクトリ上の指定ファイルについての情報を表示する。

A＞DIR_x：パス名⏎

ドライブx：の「パス名」で指定されたディレクトリ上のすべてのファイル情報を表示する。「A＞DIR_x：パス名¥*.*⏎」と同じ意味。

A＞DIR_x：パス名¥ファイル名⏎

ドライブx：の「パス名」で指定されたディレクトリ上の指定ファイルの情報を表示する。

なお、すべてのDIRコマンドにおいて、2つのスイッチ――「/W」によるワイド表示、および「/P」によるページ単位の表示が可能である。なお、各スイッチは、⏎の直前に付ける。

A＞DIR_x：/W⏎

ワイド表示。ファイルの作成年月日等の表示を省略し、1行に5個のファイル名を表示する。

A＞DIR_x：/P⏎

ディレクトリ表示のスクロールを1ページ(23行)ごとに止めて表示する。

A＞DIR_x：/W/P⏎

スイッチ「W」＋「P」の機能

［解説］　MS-DOSのコマンドのなかで、もっとも多用するコマンドである。いずれのDIRコマンドも、カレントディレクトリ、または指定ディレクトリ上に直接登録されている範囲のファイル情報を表示するもので、一度に他のディレクトリ上のファイルや階層構造などを表示することはできない。

```
A>DIR B:↵

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

COMMAND  COM     24931  88-07-13    0:00
CONFIG   SYS        87  89-07-29   17:25
PRINT    SYS      5855  88-07-13    0:00
WP           <DIR>      89-07-30   15:21
MP           <DIR>      89-07-30   15:21
MSDOS        <DIR>      89-07-30   15:21
        6 個のファイルがあります.
   749568 バイトが使用可能です.

A>
```

- A>DIR B:↵ ……ドライブB：のカレントディレクトリ(ルートディレクトリ)上のファイル情報を表示する
- B:¥ ←ルートディレクトリを示す
- COMMAND～PRINT：ファイル
- WP～MSDOS：当ディレクトリに存在するサブディレクトリ

図 6.12a　カレントディレクトリ上の全ファイルの情報を表示する

```
A>DIR B:*.SYS/W↵

 ドライブ C: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

CONFIG   SYS     PRINT    SYS
        2 個のファイルがあります.
   749568 バイトが使用可能です.

A>
```

- A>DIR B:*.SYS/W↵ ……前実行例(図6.12a)に、ワイルドカード「*.SYS」と、ワイド表示のスイッチ「/W」を使った例
- ファイルタイプが「.SYS」のファイルのみが、ワイド表示されている。ファイル名のみで、他の情報は表示されない

図 6.12b　ワイルドカードとワイド表示を使ってファイルの情報を表示する

```
A>DIR B:¥WP¥BUN↵
 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥WP¥BUN

.            <DIR>      89-07-30   15:26
..           <DIR>      89-07-30   15:26
NMSD1            14700  89-07-29   19:12
NMSD2            21455  89-07-21    1:27
EIGY05   BUN     25609  89-07-29   18:59
EIGY06   BUN     12872  89-07-29   19:00
EIGY07   BUN      8704  89-07-29   19:01
        7 個のファイルがあります.
   749568 バイトが使用可能です.

A>
```

- A>DIR B:¥WP¥BUN↵ ……ドライブB：のディレクトリ[BUN]上のファイル情報を表示する　カレントディレクトリに関係なく、指定ディレクトリ上のファイルが表示される
- . ……「自分」のディレクトリを示す
- .. ……親ディレクトリ(この場合はWPディレクトリ)を示す
- NMSD1～EIGY07：すべてのファイル(*.*)が表示されている
- このコマンドは、「A>DIR B：¥WP¥BUN¥*.*↵」と同じである

図 6.12c　パス名で指定されたディレクトリ上の全ファイルの情報を表示する

```
A>DIR B:¥WP¥BUN¥EIGYO5.BUN⏎

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥WP¥BUN

EIGYO5   BUN    25609  89-07-29  18:59
        1 個のファイルがあります.
   749568 バイトが使用可能です.

A>
```


図6.12d　パス名で指定されたディレクトリ上の指定ファイルの情報を表示する

テキストファイル(文字ファイル)をタイプアウトする

［内部コマンド　　TYPE］

A＞TYPE　x：パス名¥ファイル名⏎

ドライブx：の「パス名」で指定されたディレクトリ上のテキストファイル「ファイル名」の内容をタイプアウト(表示)する。

［解説］　DIRコマンドの次によく使われるコマンドである。テキストファイル(文字ファイル)をコンソール(画面)に表示する。CTRL＋Sの入力で、表示のスクロールを一時停止できる(再開は任意のキーの入力で)。また、CTRL＋Pの入力で、同時にプリンタにも出力される(再度のCTRL＋Pの入力でOFF。307ページの「プリントアウトに関するコマンド」を参照)。

```
A>TYPE B:CONFIG.SYS⏎
DEVICE=PRINT.SYS
DEVICE=RSDRV.SYS
DEVICE=ATOK6A.SYS /E=1 /T=1
DEVICE=ATOK6B.SYS


A>
```


図6.13　テキストファイルをタイプアウトする

ディスク上のファイルをコピーする

［内部コマンド　COPY　：　ファイル名にワイルドカードの使用可能］

A＞COPY_*x*：パス名¥ファイル名_*y*：⏎

ドライブ*x*：の「パス名」で指定されたディレクトリ上のファイル「ファイル名」を、ドライブ*y*：上のカレントディレクトリにコピーする。

A＞COPY_*x*：パス名 1¥ファイル名 1_*y*：パス名 2¥ファイル名 2⏎

ドライブ*x*：の「パス名 1」で指定されたディレクトリ上のファイル「ファイル名 1」を、ドライブ*y*：の「パス名 2」で指定されたディレクトリ上にファイル名を「ファイル名 2」としてコピーする。

A＞COPY_*x*：パス名¥ワイルドカード_*y*：/V⏎

ドライブ*x*：の「パス名」で指定されたディレクトリ上のファイルのなかで、「ワイルドカード」にマッチ(合致)するものをドライブ*y*：上のカレントディレクトリにベリファイ付きでコピーする。

［解説］　COPYコマンド全般について、上の例のような、ワイルドカードによるファイル指定ができる。また、スイッチ「/V」を⏎の直前に付けることにより、リード・アフター・ライトのベイファイ(チェック)が行われ、正しくコピーされたかどうかがチェックされる(ただし、コピー速度は1.5倍ほど遅くなる。281ページのVERIFYコマンド参照)。

COPYコマンドは、コンソール、プリンタ、ディスクファイルなどの、周辺装置間のデータ転送コマンドであるが、普通は、ディスク上のファイルのコピー(転送)コマンドとして使われる場合が多い。

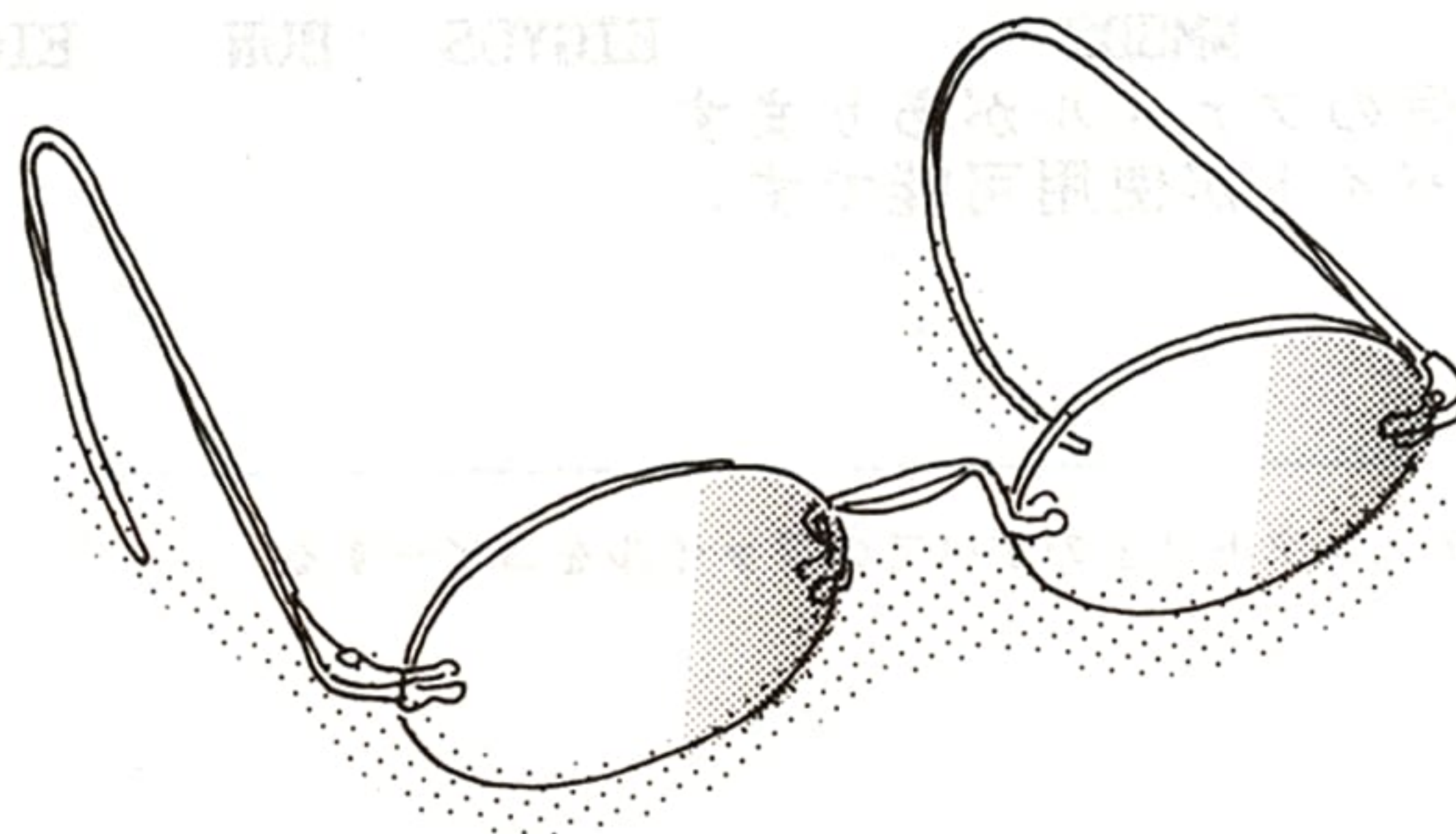
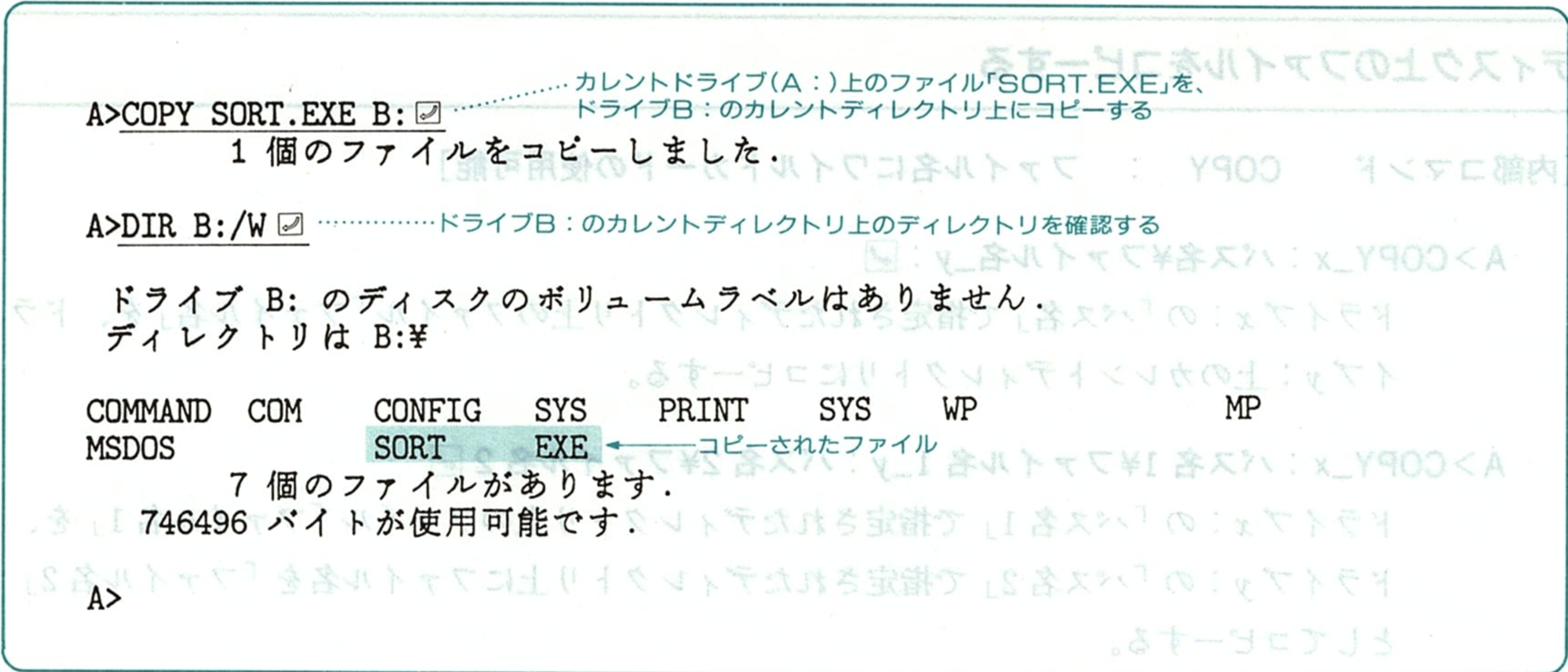
```
A>COPY SORT.EXE B:⏎
        1 個のファイルをコピーしました.

A>DIR B:/W⏎

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

COMMAND  COM     CONFIG   SYS     PRINT    SYS     WP              MP
MSDOS            SORT     EXE
        7 個のファイルがあります.
   746496 バイトが使用可能です.

A>
```


図 6.14a　カレントディレクトリ上のファイルをコピーする

```
A>COPY B:¥WP¥BUN A:⏎
B:¥WP¥BUN¥NMSD1
B:¥WP¥BUN¥NMSD2
B:¥WP¥BUN¥EIGY05.BUN
B:¥WP¥BUN¥EIGY06.BUN
B:¥WP¥BUN¥EIGY07.BUN
        5 個のファイルをコピーしました.

A>DIR/W⏎

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

COMMAND  COM   EMSDRIVE SYS   GRAPH    SYS   MOUSE    SYS   PRINT    SYS
RAMDISK  SYS   RSDRV    SYS   CONFIG   SYS   XCOPY    EXE   ASSIGN   EXE
ATTRIB   EXE   BACKUP   EXE   CHKDSK   EXE   CUSTOM   EXE   CHKFIL   EXE
DISKCOPY EXE   DUMP     EXE   EDLIN    EXE   FC       EXE   FORMAT   EXE
KEY      EXE   SETUP2   EXE   REPLACE  EXE   RESTORE  EXE   SPEED    EXE
SWITCH   EXE   SYMDEB   EXE   SYS      EXE   TREE     EXE   USKCGM   EXE
NMSD1          NMSD2          EIGY05   BUN   EIGY06   BUN   EIGY07   BUN
        35 個のファイルがあります.
   408576 バイトが使用可能です.

A>
```


図 6.14b　指定したディレクトリ上のすべてのファイルをコピーする

```
A>COPY B:¥WP¥BUN¥*.BUN A:¥WP¥JXW ⏎
B:¥WP¥BUN¥EIGYO5.BUN
B:¥WP¥BUN¥EIGYO6.BUN
B:¥WP¥BUN¥EIGYO7.BUN
        3 個のファイルをコピーしました.

A>DIR ¥WP¥JXW ⏎

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥WP¥JXW

.             <DIR>      89-08-23   18:00
..            <DIR>      89-08-23   18:00
EIGYO5   BUN     25609  89-07-29   18:59
EIGYO6   BUN     12872  89-07-29   19:00
EIGYO7   BUN      8704  89-07-29   19:01
        5 個のファイルがあります.
   358400 バイトが使用可能です.

A>
```


図 6.14c　ワイルドカードで指定したファイルを、指定ディレクトリ上にコピーする

アーカイブ属性の付いたファイルや、指定した日付以降に作成されたファイルのみをコピーする

［<3.x>外部コマンド　　XCOPY.EXE　：　ファイル名にワイルドカードの使用可能］

A>XCOPY_x：パス名 1¥ワイルドカード_y：パス名 2_/A ⏎

ドライブ x：の「パス名 1」で指定されたディレクトリ上の、「ワイルドカード」にマッチ(合致)するファイルのなかで、アーカイブ属性のついたファイル(まだバックアップコピーがされていないファイル)のみを、ドライブ y：上の「パス名 2」で指定されたディレクトリにコピーする。「パス名 2」で指定したコピー先のディレクトリがない場合には、問い合せに答えることにより、そのディレクトリが作成される。なお、「/A」の代わりに「/M」スイッチをつけた場合は、「/A」と同様のコピーを行うが、さらに、コピーしたコピー元ファイルのアーカイブ属性を解除する。

A>XCOPY_x：パス名¥ワイルドカード_y：_/D：89-09-01 ⏎

ドライブ x：の「パス名」で指定されたディレクトリ上の、「ワイルドカード」にマッチ(合致)するファイルのなかで、89 年 9 月 1 日以降に作成されたファイルのみを、ドライブ y：上のカレントディレクトリにコピーする。

なお、「/A」と「/D：」スイッチを合せて、「/A/D：89-09-01」とすることもできる。各スイッチは、↵の直前に付ける。

［解説］　XCOPY コマンドは、本来(スイッチにより)、COPY コマンドではコピーできない、任意のディレクトリ上のすべてのサブディレクトリとそのすべてのファイルを、階層構造ごとそっくりコピーすることができる。このコマンドは、MS-DOS のコマンドとしては作りが少々変則的であり、「XCOPY.EXE」を「MCOPY.EXE」にリネームし、「MCOPY コマンド」として利用することを勧める(ここでの呼び方はいちおう、XCOPY としておくが)。リネームの件、および階層構造のコピーに関しては、「階層ディレクトリに関するコマンド」の項(320ページ)で解説する。

XCOPY コマンドは、各種のスイッチを付けない場合、基本的には COPY コマンドと同じ使い方で同じ働きをするディスクファイルのコピーコマンド(ただし、ファイルの連結や、デバイスファイル間のデータ転送はできない)と考えてよいが、「/A(/M)」や「/D：」スイッチを指定すると、アーカイブ属性や、ファイルの作成日付などによる条件選択のコピーを行うことができる。したがって、ハードディスクなどのバックアップコピーにも利用できる。なお、XCOPY コマンドにはこのほかにもいくつかのスイッチによる機能があるが、ここでは省略する(また、当コマンドやアーカイブ属性などについては、4.5 章の XCOPY コマンドの項(205 ページ)でくわしく解説している)。

```
A>XCOPY ¥WP¥BUN B: /A↵

送り側のファイルを読込み中です．．．
A:¥WP¥BUN¥EIGYO5.BUN
A:¥WP¥BUN¥EIGYO6.BUN
A:¥WP¥BUN¥EIGYO7.BUN
        3 個のファイルをコピーしました．

A>DIR B:↵

 ドライブ B: のディスクのボリュームラベルはありません．
 ディレクトリは B:¥

EIGYO5   BUN     25609  89-07-29   18:59
EIGYO6   BUN     12872  89-07-29   19:00
EIGYO7   BUN      8704  89-07-29   19:01
        3 個のファイルがあります．
  1201152 バイトが使用可能です．
```

A>XCOPY ¥WP¥BUN B: /A↵ ……カレントドライブ(A：)のディレクトリ[BUN]上のファイルのなかから、アーカイブ属性のついたファイルのみをドライブB：のカレントディレクトリ上にコピーする

A>DIR B:↵ ……コピーされたファイルを確認する

EIGYO5～EIGYO7 ……この3つのアーカイブファイルがコピーされた

```
A>ATTRIB B:*.*⏎ ……………コピーされたファイルのアトリビュート(属性)を確認する。
                       (ATTRIBコマンドについては、297ページを参照)
  A       B:¥EIGYO5.BUN
  A       B:¥EIGYO6.BUN
  A       B:¥EIGYO7.BUN
  ↑
  └────確かにアーカイブ属性のついたファイルのみがコピーされている
A>
```

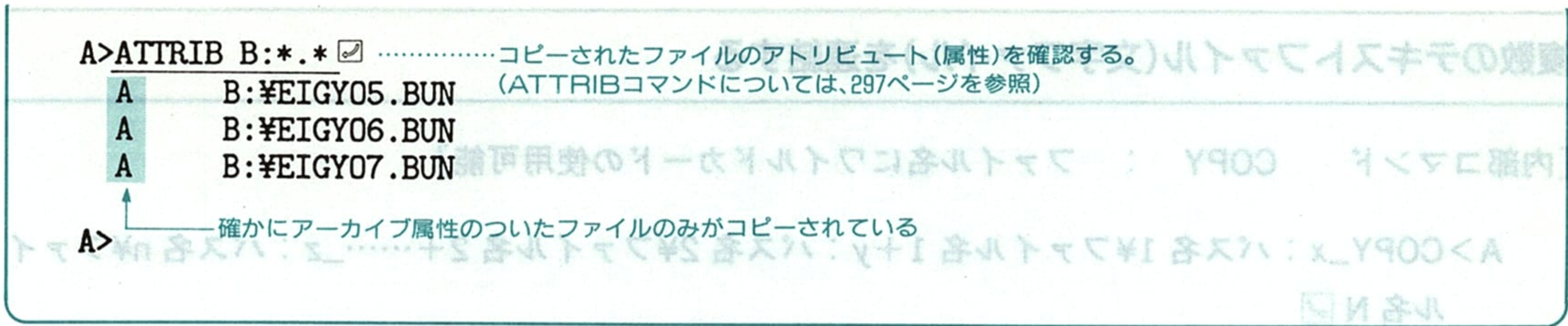
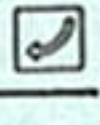
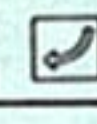



図 6.15a　アーカイブ属性のついたファイルのみをコピーする

```
A>DIR ¥WP¥BUN⏎ ……………カレントドライブ(A:)のディレクトリ[BUN]上のファイルを確認する

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥WP¥BUN

.            <DIR>      89-07-20   15:26
..           <DIR>      89-07-20   15:26
NMSD1           14700   89-08-25   19:12
NMSD2           21455   89-07-21    1:27
EIGYO5   BUN    25609   89-08-10   17:59
EIGYO6   BUN    12872   89-07-23   19:21
EIGYO7   BUN     8704   89-07-29   19:01
        7 個のファイルがあります.
   749568 バイトが使用可能です.

A>XCOPY ¥WP¥BUN B: /D:89-07-29⏎ ……カレントドライブ(A:)のディレクトリ[BUN]上のファイルのなか
                                    から、89年7月29日以降に作成されたファイルのみをドラ
                                    イブB:のカレントディレクトリ上にコピーする

送り側のファイルを読込み中です...
A:¥WP¥BUN¥NMSD1
A:¥WP¥BUN¥EIGYO5.BUN
A:¥WP¥BUN¥EIGYO7.BUN
        3 個のファイルをコピーしました.

A>DIR B:⏎ ……………ドライブB:のディレクトリを確認する

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥
                              ┌──この日付で選択される
                              ↓
EIGYO5   BUN    25609   89-08-10   17:59 ┐
EIGYO7   BUN     8704   89-07-29   19:01 │89年7月29日以降に作成された
NMSD1           14700   89-08-25   19:12 ┘ファイルがコピーされている
        3 個のファイルがあります.
  1199104 バイトが使用可能です.

A>
```

図 6.15b　特定の日付以降のファイルのみをコピーする

複数のテキストファイル(文字ファイル)を連結する

[内部コマンド　COPY　:　ファイル名にワイルドカードの使用可能]

A>COPY_*x*:パス名1¥ファイル名1+*y*:パス名2¥ファイル名2+……_*z*:パス名n¥ファイル名N⏎

ドライブ*x*:、*y*:、…のそれぞれの「パス名」で指定されるディレクトリ上のファイル「ファイル名1」、「ファイル名2」、…を連結し、ドライブ*z*:の「パス名n」で指定されたディレクトリ上に1つのファイル「ファイル名N」としてコピーする。

[解説]　COPYコマンドの連結機能を使う。連結するファイル名を「+」記号でつなげてコマンドラインに記述する。通常のテキストファイルの連結は、このコマンド形式でよいが、ファイルの終了コードの「$1A_H$」が、ファイルの途中に含まれる特殊なテキストファイルや、マシン語ファイルなどの連結には、それらのファイル名のあとに、スイッチ「/B」を付けなければならない。

```
A>COPY README.DOC+B:CONFIG.SYS B:MERGE⏎ ····カレントドライブ(A:)上のファイル「README.DOC」と、ドライブB:上のファイル「CONFIG.SYS」を連結したファイル「MERGE」をドライブB:上に作成する(いずれも文章ファイルの場合)
README.DOC
B:CONFIG.SYS
        1 個のファイルをコピーしました.

A>DIR B:MERGE⏎ ················ドライブB:上に作成されたファイル「MERGE」の確認

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

MERGE            10664  89-08-23   18:13 ···············連結されたファイルが作成されている
        1 個のファイルがあります.
   738304 バイトが使用可能です.

A>
```

図6.16a　COPYコマンドにより、複数のファイルを連結する

```
A>TYPE B:MERGE ⏎ ……………連結されたファイルの内容をタイプアウトして確認する
 1.    バックアップ作成時の注意
                              「README.DOC」ファイル

  ＰＳ９８－０１３（０１４）－ＨＵ／Ｈ４Ｗのシステムディスクは、９セクタ
 フォーマット（７２０ＫＢ）でフォーマットされています。
  ＰＳ９８－０１３（０１４）－ＨＵ／Ｈ４Ｗのバックアップを作成する時には、
                         （途中省略）
  なお、「スキップセクタ」が表示された場合、使用可能ディスク容量を、本製品
 に添付されている「固定ディスクを使用する場合のご注意」で確認してください。
  使用可能ディスク容量が記載されている値以上であれば、正常にご使用いただけ
 ます。
DEVICE=PRINT.SYS                                          ↕ 連結されている
DEVICE=RSDRV.SYS
DEVICE=ATOK6A.SYS /E=1 /T=1        「CONFIG.SYS」ファイル
DEVICE=ATOK6B.SYS

A>
```

図 6.16b 連結されたファイルを確認する

テキストファイル(文字ファイル)を作成する

［内部コマンド　COPY］ または ［外部プログラム　EDLIN.EXE］

A＞COPY_CON_*x*：パス名¥ファイル名⏎

キー入力したテキストを、ドライブ*x*：の「パス名」で指定されたディレクトリ上にファイル「ファイル名」としてセーブする。

［解説］ この方法は、簡単な(短い)ファイルの作成によく使われ、本書でも随所で利用している(215ページの5.1章参照)。COPYコマンドを使って、コンソールの「ファイル」(つまりキー入力文字)をディスク上に転送(コピー)することにより、ファイルが作成される。キー入力テキストの最後には、CTRL＋Zを入力し、⏎する。

```
A>COPY CON B:TEXT ⏎          ……COPYコマンドを使って、ドライブB：上にキー入力による
                                テキストファイル「TEXT」を作成する
ＣＯＰＹコマンドを使って，⏎
キー入力文字によるファイル「ＴＥＸＴ」を⏎      各キー入力行の最後は⏎を入力する
ドライブＢ：上に作成します．⏎
^Z ⏎ ……… CTRL+Z ⏎
        1 個のファイルをコピーしました．

A>DIR B:TEXT ⏎ ……作成されたドライブB：上のファイル「TEXT」を確認する

 ドライブ B: のディスクのボリュームラベルはありません．
 ディレクトリは B:¥

TEXT            100  89-08-23   16:05 ……ファイルは作成されている
        1 個のファイルがあります．
  745472 バイトが使用可能です．

A>TYPE B:TEXT ⏎  ……ドライブB：上に作成されたファイル「TEXT」の
                    内容をタイプアウトして確認する
ＣＯＰＹコマンドを使って，
キー入力文字によるファイル「ＴＥＸＴ」を      キー入力したものがファイルになっている
ドライブＢ：上に作成します．

A>
```

図6.17　COPYコマンドにより文字ファイルを作成する

A＞EDLIN_*x*：パス名¥ファイル名⏎

ドライブ*x*：の「パス名」で指定されたディレクトリ上に、新しいファイル「ファイル名」を作成する。もし、「ファイル名」が既存ファイルであれば、そのファイルの編集作業となる。

［解説］　MS-DOSのシステムディスクには、「EDLIN.EXE」というエディタプログラムが含まれている。これはスクリーンエディタではなく、ラインエディタと呼ばれるエディタで、能率はよくないが、ファイルの作成／編集を自由に行うことができる。くわしくは5.2章(221ページ)を参照。

ファイルを削除する

［内部コマンド　DEL（また ERASE）　：　ファイル名にワイルドカードの使用可能］

A＞DEL_*x*：パス名¥ファイル名⏎

ドライブx：の「パス名」で指定されたディレクトリ上のファイル「ファイル名」を削除する。

A＞DEL_*x*：パス名⏎

ドライブx：の「パス名」で指定されたディレクトリ上に直接存在するすべてのファイルを削除する（指定ディレクトリ上のすべてのサブディレクトリや、サブディレクトリ上のファイルには影響しない）。これはコマンド「**A＞DEL_*x*：パス名¥*.*⏎**」と同じである。

［解説］　コマンドは、「DEL」または「ERASE」のどちらでも実行可能。DEL コマンドによるファイルの削除は、カレントディレクトリ、あるいは指定ディレクトリに直接存在するファイルが対象となり、そのほかのディレクトリ上のファイルには影響しない。

```
A>DEL B:TEXT ⏎ ……………前実行例（図6.17）で作成した、ドライブB：上のファイル「TEXT」を削除する

A>DIR B:TEXT ⏎ ……………削除したファイルの情報を見る

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

ファイルが見つかりません. ……………当然のことながら存在していない

A>
```

図 6.18a　カレントディレクトリ上のファイルを削除する

```
A>DIR ⏎ ……………カレントドライブ（A：）上のディレクトリを見る

NMSD1              14700  89-07-29   19:12
NMSD2              21455  89-07-21    1:27
EIGY05    BUN      25609  89-07-29   18:59
EIGY06    BUN      12872  89-07-29   19:00
EIGY07    BUN       8704  89-07-29   19:01
BUNSHO        <DIR>       89-08-23   18:00
       36 個のファイルがあります.
   358400 バイトが使用可能です.
```

```
A>DEL EIGYO*.*         ……………これらのファイルから、先頭の5文字が「EIGYO」であるすべてのファイルを削除する

A>DIR         ……………実行後のディレクトリを確認する

NMSD1              14700  89-07-29   19:12
NMSD2              21455  89-07-21    1:27
BUNSHO        <DIR>       89-08-23   18:00   ←ここにあった3つのファイルが削除されている
       33 個のファイルがあります.
   407552 バイトが使用可能です.

A>
```

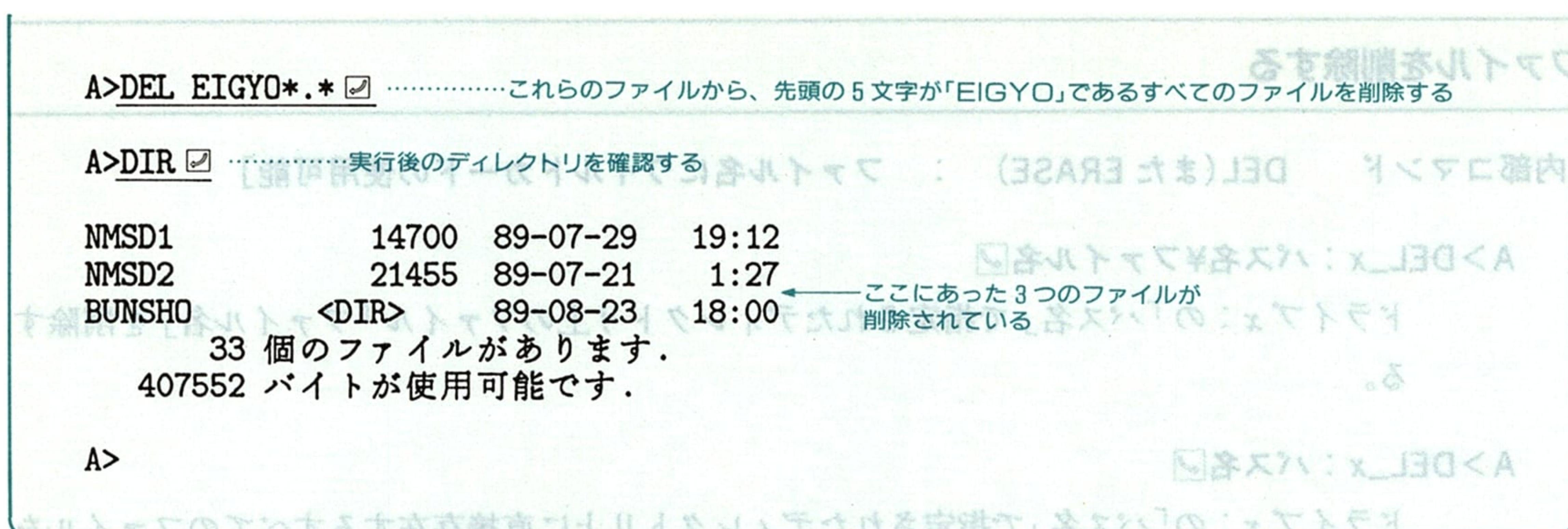



図6.18b　ワイルドカードを使ってファイルを削除する

```
A>DIR ¥WP¥BUN /W         ……………カレントドライブ(A:)のディレクトリ[¥WP¥BUN]上のファイルを確認する

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥WP¥BUN
                                         3つのファイルが存在している
.            ..              EIGYO5   BUN    EIGYO6   BUN    EIGYO7   BUN
        5 個のファイルがあります.
   407552 バイトが使用可能です.
                             このディレクトリ[¥WP¥BUN]上のすべてのファイルを削除する
A>DEL ¥WP¥BUN                (ワイルドカードによる指定ではないことに注目)
よろしいですか <Y/N>? Y      ……………確認の問い合わせがある

A>DIR ¥WP¥BUN /W         ……………実行後のディレクトリ[¥WP¥BUN]を確認する

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥WP¥BUN

.            ..                       ファイルは存在していない
        2 個のファイルがあります.
   456704 バイトが使用可能です.

A>
```

図6.18c　指定ディレクトリ上に直接存在するすべてのファイルを削除する

ファイル名を変更する

［内部コマンド　REN　：　ファイル名にワイルドカードの使用可能］

A>REN_x：パス名¥ファイル名 1_ファイル名 2

ドライブ x：の「パス名」で指定されたディレクトリ上のファイル「ファイル名 1」を「ファイル名 2」に変更する。

［解説］　もし、変更後の新しい名前「ファイル名 2」と同一の名前のファイルが、同じディレクトリ上にすでに存在している場合、エラーメッセージが表示され、変更は行われない。

```
A>DIR /W ……………カレントドライブ(A：)上のディレクトリを見る

COMMAND  COM     ASSIGN   EXE     ATTRIB   EXE     BACKUP   EXE     CHKDSK   EXE
CHKFIL   EXE     CUSTOM   EXE     DISKCOPY EXE     DUMP     EXE     EDLIN    EXE
        30 個のファイルがあります.
   494592 バイトが使用可能です.

A>REN EDLIN.EXE EDITOR.EXE ……………ファイル「EDLIN.EXE」を「EDITOR.EXE」にリネームする

A>DIR /W ……………実行後のディレクトリを確認する

COMMAND  COM     ASSIGN   EXE     ATTRIB   EXE     BACKUP   EXE     CHKDSK   EXE
CHKFIL   EXE     CUSTOM   EXE     DISKCOPY EXE     DUMP     EXE     EDITOR   EXE
        30 個のファイルがあります.
   494592 バイトが使用可能です.                          ファイル名が変更されている

A>
```

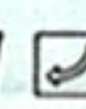



図 6.19a　カレントディレクトリ上のファイル名を変更する

```
A>DIR B:¥MSDOS/W ……………ドライブB：のディレクトリ[MSDOS]上のファイルを確認しておく

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥MSDOS

.                 ..                FORMAT   EXE     SYS      EXE     DISKCOPY EXE
CHKDSK   EXE     TOOL
         7 個のファイルがあります.
   738304 バイトが使用可能です.
```

```
A>REN B:¥MSDOS¥SYS.EXE SYSCOPY.EXE ⏎ ……任意のカレントディレクトリから、ディレクトリ[MSDOS]
                                        上のファイル「SYS.EXE」を「SYSCOPY.EXE」に
                                        リネームする
A>DIR B:¥MSDOS/W ⏎ ……実行後のディレクトリを確認する

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥MSDOS

.                 ..                FORMAT   EXE    SYSCOPY  EXE    DISKCOPY EXE
CHKDSK   EXE      TOOL
        7 個のファイルがあります.                      ↑
  738304 バイトが使用可能です.                   リネームされている

A>
```

図6.19b　指定ディレクトリ上のファイル名を変更する

```
A>DIR *.BUN/W ⏎ ……カレントドライブ(A:)上の「.BUN」ファイルを確認する

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥WP¥BUN

EIGYO5   BUN    EIGYO6   BUN    EIGYO7   BUN   3つの「.BUN」ファイルが存在している
        3 個のファイルがあります.
  749568 バイトが使用可能です.

A>REN *.BUN *.MEM ⏎ ……すべての「.BUN」ファイルを「.MEM」ファイルにリネームする

A>DIR *.MEM/W ⏎ ……実行後のディレクトリの確認

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥WP¥BUN

EIGYO5   MEM    EIGYO6   MEM    EIGYO7   MEM   「.BUN」→「.MEM」にリネームされている
        3 個のファイルがあります.
  749568 バイトが使用可能です.
A>
```

図6.19c　ワイルドカードを使ってファイル名を変更する

ファイルを更新・削除禁止／可能にする。アーカイブ属性を設定／解除する。また現在のそれらの状態を表示する

［＜3.*x*＞外部プログラム　ATTRIB.EXE　：　ファイル名にワイルドカードの使用可能］

A＞ATTRIB_＋R_*x*：パス名¥ファイル名⏎

ドライブ*x*：の「パス名」で指定されたディレクトリ上のファイル「ファイル名」をリードオンリー（更新／削除禁止）ファイルにする。

A＞ATTRIB_－R_*x*：パス名¥ファイル名⏎

前記コマンドの逆の操作。設定されているリードオンリーのアトリビュートを解除し、通常の（更新／削除可能）ファイルにする。

A＞ATTRIB_＋A_*x*：パス名¥ファイル名⏎
A＞ATTRIB_－A_*x*：パス名¥ファイル名⏎

ドライブ*x*：の「パス名」で指定されたディレクトリ上のファイル「ファイル名」のアーカイブ属性を設定（「＋」スイッチの場合）、解除（「－」スイッチの場合）する。

A＞ATTRIB_*x*：パス名¥ファイル名⏎

「パス名¥ファイル名」で指定したファイルのアトリビュートが、リードライトかリードオンリーか、アーカイブ属性が設定されているかどうかを調べる。

［解説］　MS-DOSのファイルには、システムファイル、リードオンリーファイル、アーカイブファイル（193ページの4.5章参照）、隠しファイルなど、6種類ほどのアトリビュート（属性：ファイルの性質を決定するための付加情報）を付加することができる。そのなかのリードオンリーとアーカイブのアトリビュートを付加したり解除したりするものが、当プログラム「ATTRIB」である。

リードオンリーファイル（リードオンリーのアトリビュートが付加されたファイル）は、DELコマンドが無効であり、各種のビジネスプログラムなどからも、そのファイルの更新／削除ができなくなる。したがって、書き換えられたり削除されては困るファイルに対して、ファイルを保護するために使用する。ただし、RENコマンド（リネーム）は有効なので、エディタや各種のビジネスプログラムにより更新されてしまう可能性がある（ただし、削除はできないので、リネームされても「.BAK」ファイルなどで必ず残る）。また、COPYコマンドでコピーされたコピー先のファイルは、もとがリードオンリーであっても、それが解除されているので注意を要する。

アーカイブ属性は、BACKUPコマンドでファイルをバックアップした場合に解除される。

通常、作成したファイルやCOPYコマンドでコピーされたコピー先のファイル、バックアップ以後変更があったファイルはアーカイブ属性が付けられている。それを利用して、「/M」スイッチ付きのXCOPYコマンドやBACKUPコマンドで新規に作成したファイルや変更があったファイルのみをコピーまたはバックアップすることができる。当プログラム「ATTRIB」を使えば、そのアーカイブ属性の設定／解除が自由に行える。

```
A>DIR B:↵ ……………ドライブB：上のディスクのファイルを確認しておく

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

COMMAND  COM    24931  88-07-13    0:00*
ASSIGN   EXE    26450  88-07-13    0:00
ATTRIB   EXE     9154  88-07-13    0:00
                    ・
                    ・
                    ・
PRINT    SYS     5855  88-07-13    0:00
RSDRV    SYS     7352  88-07-13    0:00
README   DOC    10578  88-07-13    0:00*
CONFIG   SYS       37  89-08-23   21:34
       35 個のファイルがあります.
    455680 バイトが使用可能です.
```

（これはバージョン3.3のMS-DOSシステムディスク。＊印の2つのファイルをリードオンリーファイルにしてみる）

```
A>ATTRIB +R B:COMMAND.COM↵ …………ドライブB：上のファイル「COMMAND.COM」をリードオンリーにする

A>ATTRIB +R B:*.DOC↵ …………ドライブB：上のファイルタイプが「.DOC」であるすべてのファイルをリードオンリーにする

A>ATTRIB B:COMMAND.COM↵ …………ドライブB：上のファイル「COMMAND.COM」がリードオンリーかどうか調べる
R A      B:¥COMMAND.COM …………先頭の「R」がリードオンリーファイルであることを示す
                                （リードライトの場合はこの部分が空白となる）

A>DEL B:*.*↵ …………ドライブB：上のすべてのファイルを削除する
よろしいですか <Y/N>? Y↵

A>DIR B:↵ …………すべてのファイルが削除されたドライブB：上のファイルを確認する

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

COMMAND  COM    24931  88-07-13    0:00 ┐リードオンリーファイルは削除されずに残っている
README   DOC    10578  88-07-13    0:00 ┘…「.DOC」はもともと1つしか存在していなかった
        2 個のファイルがあります.
   1118208 バイトが使用可能です.

A>
```

（リードオンリーを解除するには、上記と同じコマンドで、「+R」を「−R」として実行すればよい）

図6.20a　ファイルをリードオンリーにする

```
A>ATTRIB +A *.BUN⏎ ……………カレントディレクトリ上の「*.BUN」ファイルのアーカイブ属性を設定する
A>ATTRIB *.BUN⏎ ……………カレントディレクトリ上の「*.BUN」ファイルのアトリビュート(属性)を確認する
  A      A:¥WP¥BUN¥EIGY05.BUN
  A      A:¥WP¥BUN¥EIGY06.BUN
  A      A:¥WP¥BUN¥EIGY07.BUN
  └───アーカイブ属性が設定された
A>ATTRIB -A *.BUN⏎ ……………設定されたアーカイブ属性を解除する
A>ATTRIB *.BUN⏎ ……………アトリビュートを確認する
         A:¥WP¥BUN¥EIGY05.BUN
         A:¥WP¥BUN¥EIGY06.BUN
         A:¥WP¥BUN¥EIGY07.BUN
A>└───アーカイブ属性が解除された
```

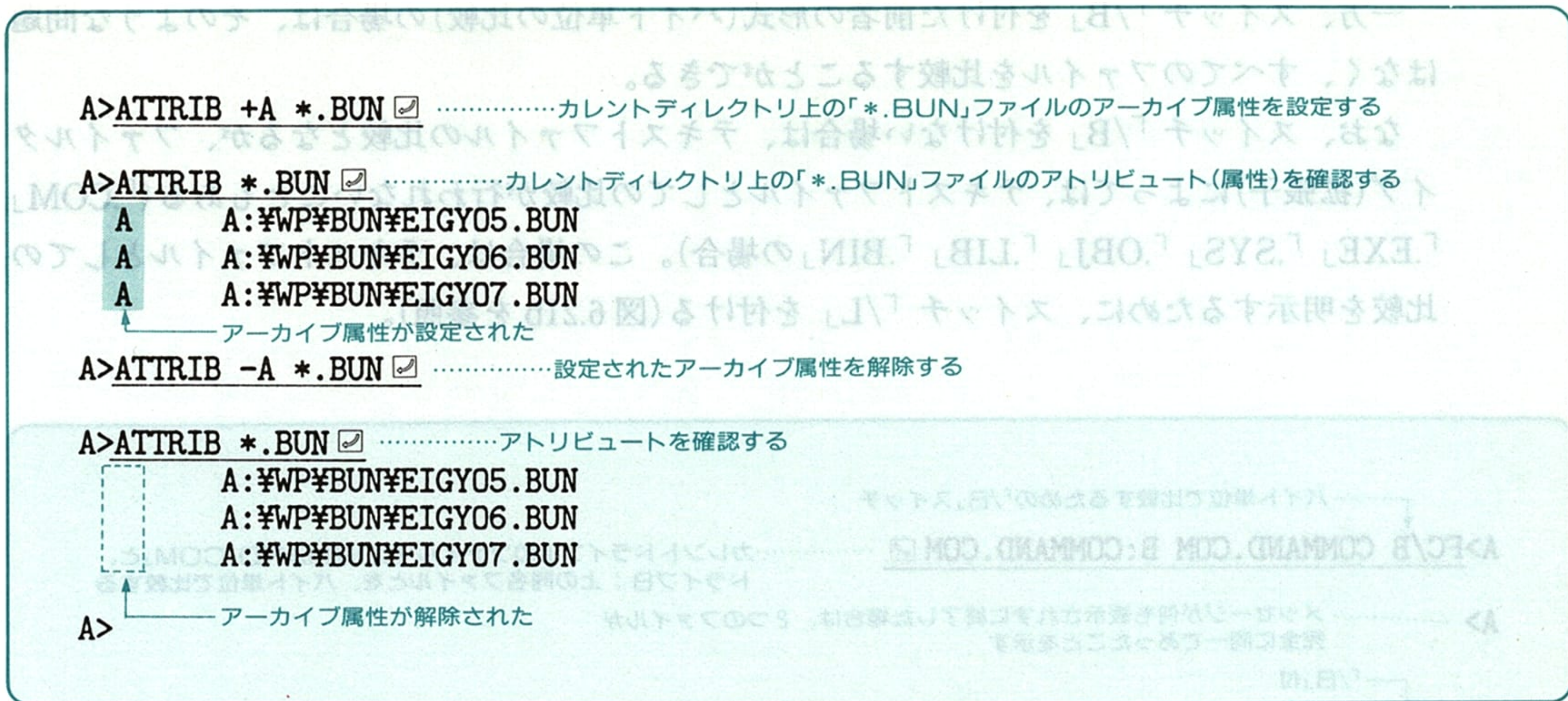



図 6.20b　ファイルのアーカイブ属性を設定／解除する

2つのファイルの内容を比較する

[外部プログラム　FC.EXE]

A>FC/B_x：パス名 1¥ファイル名 1_y：パス名 2¥ファイル名 2⏎

ドライブx：の「パス名 1」で指定されたディレクトリ上のファイル「ファイル名 1」と、ドライブy：の「パス名 2」で指定されたディレクトリ上の「ファイル名 2」とを1バイト単位で比較する。

A>FC_x：パス名 1¥ファイル名 1_y：パス名 2¥ファイル名 2⏎

ドライブx：の「パス名 1」で指定されたディレクトリ上のファイル「ファイル名 1」と、ドライブy：の「パス名 2」で指定されたディレクトリ上の「ファイル名 2」の、2つのテキストファイル(文字ファイル)を行単位(キャリッジリターンからキャリッジリターンまでのブロック単位)で比較する。

[解説]　FCにはスイッチによるさらに多くの機能があるがここでは省略する。スイッチ「/B」を付けない後者の形式(テキストファイルをブロック単位で比較)の場合、純粋なテキストファイルでないファイル(アスキーコード以外のコードが含まれている場合)の比較は、正しく行われない場合がある。またこの形式の場合、コンピュータのメモリ内に一度に収容できない大きなファイルの場合は、全ファイルの比較ができない。

一方、スイッチ「/B」を付けた前者の形式(バイト単位の比較)の場合は、そのような問題はなく、すべてのファイルを比較することができる。

なお、スイッチ「/B」を付けない場合は、テキストファイルの比較となるが、ファイルタイプ(拡張子)によっては、テキストファイルとしての比較が行われないこともある(「.COM」「.EXE」「.SYS」「.OBJ」「.LIB」「.BIN」の場合)。この場合は、テキストファイルとしての比較を明示するために、スイッチ「/L」を付ける(図6.21bを参照)。

```
A>FC/B COMMAND.COM B:COMMAND.COM

A>

A>FC/B COMMAND.COM B:COMMAND.COM
00000DCF: 0A 40
00000F4B: 0A 40
00000F4F: A0 00
00000F50: 00 04
0000290D: FE 08
0000290E: CB DB
0000290F: 74 75
^C

A>
```


図6.21a　バイト単位で2つのファイルの内容を比較する

```
A>FC /L CONFIG.SYS B:CONFIG.SYS

***** CONFIG.SYS
DEVICE=NECAIK1.DRV
DEVICE=NECAIK2.DRV B:NECAI.SYS
***** B:CONFIG.SYS
DEVICE=ATOK6A.SYS /E=1 /T=1
DEVICE=ATOK6B.SYS

*****

A>
```


図6.21b　行単位で、2つの文字ファイルの内容を比較する

コマンド実行の表示出力や、ファイルの内容をソートする

［外部プログラム　　SORT.EXE］

A＞コマンド_|_SORT⏎

「コマンド」(DIR や CHKDSK 等)の表示出力をパイプ「|」で接続し、ABC 順にソートして表示する。

A＞SORT_＜_x：パス名¥ファイル名⏎

ドライブ x：の「パス名」で指定されたディレクトリ上のファイル「ファイル名」の内容をソートして表示する。

「SORT」には、逆順ソートする「/R」、入力データの各行の n 文字目からをソートの対象にする「/＋n」の、2 つのスイッチがある。「/R」スイッチの一例を示す。

A＞コマンド_|_SORT/R⏎

上記の最初のコマンドの逆順ソート

A＞SORT/R_＜_x：パス名¥ファイル名⏎

同 2 番目のコマンドの逆順ソート。

なお、これらの実行結果の表示は、リダイレクト記号「＞」を使えば、ファイルにセーブすることができる。その一例を示す。

A＞コマンド_|_SORT＞x：パス名¥ファイル名⏎

最初のコマンド例の結果を、ドライブ x：の「パス名」で指定されたディレクトリ上のファイル「ファイル名」としてセーブする。

［解説］　SORT コマンドは「フィルタ」であり、この SORT フィルタへのデータ入力は、パイプまたはリダイレクトでのみ可能である。なお、SORT コマンドの実行には、内部的にカレントドライブの読み／書きが行われるため、そのフロッピーディスクなどが書き込み禁止であれば実行できないので注意すること。

```
A>DIR B:¥MSDOS¥*.EXE ↵

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥MSDOS

FORMAT   EXE    97766  88-07-13    0:00
SYSCOPY  EXE    25480  88-07-13    0:00
DISKCOPY EXE    19604  88-07-13    0:00
CHKDSK   EXE    10384  88-07-13    0:00
        4 個のファイルがあります.
   738304 バイトが使用可能です.

A>DIR B:¥MSDOS¥*.EXE | SORT ↵

        4 個のファイルがあります.
   738304 バイトが使用可能です.
 ディレクトリは B:¥MSDOS
 ドライブ B: のディスクのボリュームラベルはありません.
CHKDSK   EXE    10384  88-07-13    0:00
DISKCOPY EXE    19604  88-07-13    0:00
FORMAT   EXE    97766  88-07-13    0:00
SYSCOPY  EXE    25480  88-07-13    0:00

A>
```


図6.22a　DIRコマンドの表示出力をソートする

```
A>DIR B:¥MSDOS¥*.EXE | SORT /R/+13 ↵
 ディレクトリは B:¥MSDOS
   738304 バイトが使用可能です.
        4 個のファイルがあります.
 ドライブ B: のディスクのボリュームラベルはありません.
FORMAT   EXE    97766  88-07-13    0:00
SYSCOPY  EXE    25480  88-07-13    0:00
DISKCOPY EXE    19604  88-07-13    0:00
CHKDSK   EXE    10384  88-07-13    0:00

A>
```


図6.22b　上のソートを13文字目から行う

```
A>SORT < B:CONFIG.SYS⏎ ……………ドライブB：上のCONFIG.SYSファイルの各行を
                                ABC順にソートしてタイプアウトする

DEVICE=ATOK6A.SYS /E=1 /T=1
DEVICE=ATOK6B.SYS            各行がABC順にソートされている
DEVICE=PRINT.SYS             このファイルのもとの状態は284ページの図6.13を参照
DEVICE=RSDRV.SYS

A>
```

図 6.22c　ディスク上の文字ファイルの各行をソートする

文字ファイルから指定した文字列をサーチし、その行を表示する

［外部プログラム　　FIND.EXE］

A＞FIND_〝文字列〟_x：パス名 1¥ファイル名 1_y：パス名 2¥ファイル名 2……⏎

指定した「文字列」を、ドライブ x：、y：…の、それぞれの「パス名」で指定されたディレクトリ上のファイル、「ファイル名 1」、「ファイル名 2」…から検索し、その文字列が存在する行のすべてを表示する。

A＞コマンド_|_FIND_〝文字列〟⏎

「コマンド」(DIR や CHKDSK 等)の表示出力をパイプ「|」で接続し、そのなかから指定した「文字列」を検索し、存在する行のすべてを表示する。

「FIND」に、スイッチ「/V」を付けると、検索する文字列を含まない行のすべてを表示する。そのコマンドの一例を示す。なおそのほかのスイッチは省略する。

A＞FIND/V_〝文字列〟_x：パス名 1¥ファイル名 1_y：パス名 2¥ファイル名 2……⏎

上記の最初のコマンドで、検索する文字列を含まない行のすべてを表示する。

A＞コマンド_|_FIND/V_〝文字列〟⏎

上記の 2 番目のコマンドで、検索する文字列を含まない行のすべてを表示する。

［解説］　FIND は、プログラムとしても、フィルタとしても実行可能である。最初のコマンド例はプログラムとして、2 番目の例はフィルタとして実行する場合である。いずれの場合もリダイレクト記号「＞」を使えば、実行結果の表示をファイルにセーブすることができる。

```
A>FIND "サンプル" README.DOC

---------- README.DOC
  5．3．アプリケーション登録用定義ファイルのサンプルについて
  plan用のサンプルファイルを用いていますが、システムディスク内にはMP.
  INIの他に以下のサンプルファイルが提供されています。
      OFG．INI    ・・・  OfficeGraph用サンプルファイル
      TELE．INI   ・・・  Teletool用サンプルファイル

A>
```

複数のファイルを指定してもよい

カレントドライブ(A：)上のファイル「README.DOC」から、文字列「サンプル」をサーチし、それを含む行をすべて表示する

表示されている行が文字列「サンプル」を含むすべての行である

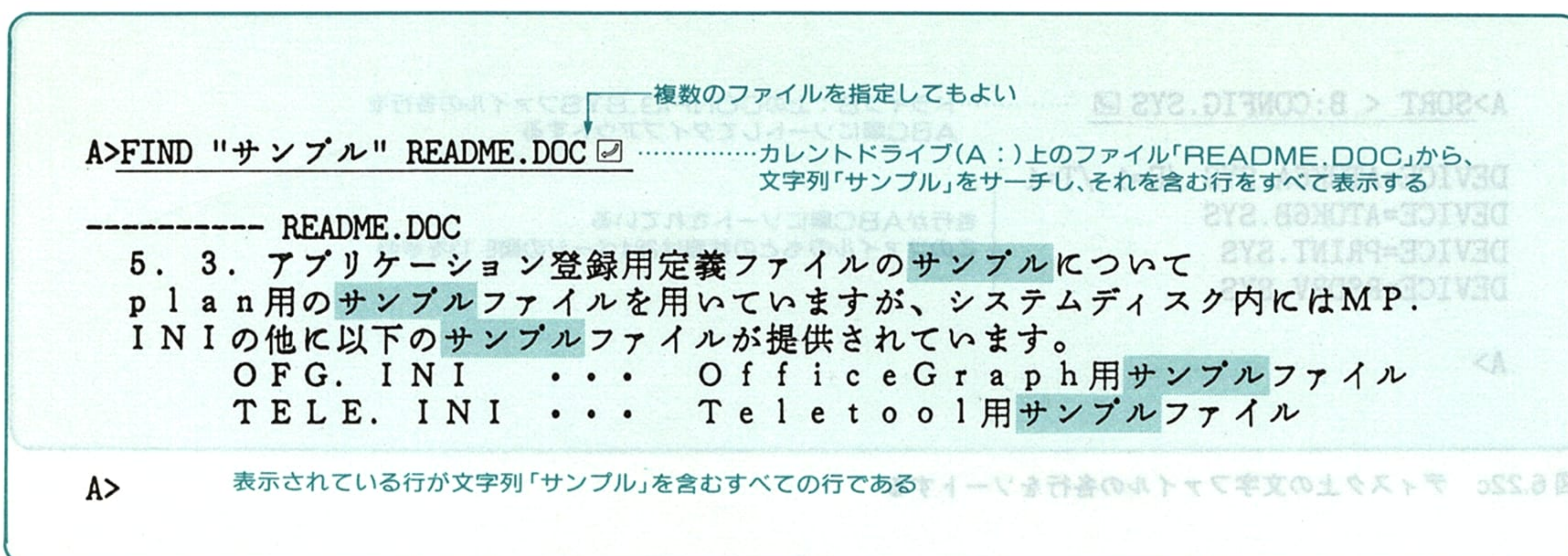



図 6.23a　文字ファイルの内容から指定した文字列を検索する

```
A>DIR B: | FIND "88-"
COMMAND  COM    24931  88-07-13    0:00
PRINT    SYS     5855  88-07-13    0:00

A>DIR B: | FIND/V "88-"

 ドライブ B: のディスクのボリュームラベルはありません．
 ディレクトリは B:¥

CONFIG   SYS       87  89-07-29   17:25
WP           <DIR>     89-07-30   15:21
MP           <DIR>     89-07-30   15:21
MSDOS        <DIR>     89-07-30   15:21
        6 個のファイルがあります．
   749568 バイトが使用可能です．

A>
```

ドライブB：上のディレクトリから、88年に作成されたすべてのファイルを表示する。つまり文字列「88－」をサーチする

88年以外の年に作成されたファイルを表示する

「88－」は含まれていない。ただし、年月日の「年」以外にも、文字列「88－」が含まれていると表示されないことになる

図 6.23b　DIR コマンドの表示出力から指定のファイルを検索する

SORT(並べ替え)やFIND(検索)の結果をファイルにセーブする

［内部コマンド(機能)　　リダイレクトの「>」および「<」］

A＞DIR_|_SORT_＞_*x*：パス名¥ファイル名

DIR コマンドによる表示出力をソートし、その結果をドライブ*x*：の「パス名」で指定されたディレクトリ上に「ファイル名」という名でセーブする。

A>SORT_<_*x*：パス名 1¥ファイル名 1_>_*y*：パス名 2¥ファイル名 2

ドライブx：の「パス名 1」で指定されたディレクトリ上の「ファイル名 1」の各行をソートし、その結果をドライブy：の「パス名 2」で指定されたディレクトリ上の「ファイル名 2」という名でディスクにセーブする。

A>FIND_〝文字列〟_<_*x*：パス名 1¥ファイル名 1_>_*y*：パス名 2¥ファイル名 2

ドライブx：の「パス名 1」で指定されたディレクトリ上のファイル「ファイル名 1」から〝文字列〟を捜し出し、それが含まれるすべての行をドライブy：の「パス名 2」で指定されたディレクトリ上の「ファイル名 2」という名でディスクにセーブする。

［解説］　テキストファイルや、各種のコマンド実行による表示出力をフィルタで処理(ソートや検索)し、その結果をリダイレクト記号の「>」を使ってディスクにセーブすることができる(SORT、FIND については前述)。なお「>>」記号を使えば、既存ファイルに継ぎ足していくことも可能である。

```
A>DIR *.EXE | SORT/+13 > DIRSIZE

A>TYPE DIRSIZE

FIND      EXE     6573  88-07-13    0:00
TREE      EXE     9608  88-07-13    0:00
FC        EXE    15302  88-07-13    0:00
DISKCOPY  EXE    19604  88-07-13    0:00
USKCGM    EXE    22444  88-07-13    0:00
KEY       EXE    32806  88-07-13    0:00
DUMP      EXE    40668  88-07-13    0:00
FORMAT    EXE    97766  88-07-13    0:00
 ドライブ A: のディスクのボリュームラベルはありません.
         8 個のファイルがあります.
   475136 バイトが使用可能です.
 ディレクトリは A:¥

A>
```


図 6.24a　DIR コマンドの結果をソートしてファイルにセーブする

このコマンドの実行結果を→このファイルを作成してセーブする

```
A>SORT < DIRSIZE > DIRNAME
```

先の実行例で作成されたファイル「DIRSIZE」の内容を、それぞれのファイル名のABC順にソートし、その結果をファイル「DIRNAME」を作成してセーブする

```
A>TYPE DIRNAME
```

作成されたファイルの内容をタイプアウトして確認する

```
   8 個のファイルがあります.
 475136 バイトが使用可能です.
ディレクトリは A:¥
ドライブ A: のディスクのボリュームラベルはありません.
DISKCOPY EXE    19604  88-07-13    0:00
DUMP     EXE    40668  88-07-13    0:00
FC       EXE    15302  88-07-13    0:00
FIND     EXE     6573  88-07-13    0:00
FORMAT   EXE    97766  88-07-13    0:00
KEY      EXE    32806  88-07-13    0:00
TREE     EXE     9608  88-07-13    0:00
USKCGM   EXE    22444  88-07-13    0:00

A>
```

ファイル名がABC順にソートされている

このような内容のファイル「DIRNAME」が作成されている

図6.24b　ファイルの内容をソートしてファイルにセーブする

このコマンドの実行結果を→このファイルを作成してセーブする

```
A>FIND "88-07" < DIRNAME >DIR88-07
```

先の実行例で作成されたファイル「DIRNAME」の内容から、文字列"88-07"が含まれるすべての行を捜し出し、その結果をファイル「DIR88-07」を作成してセーブする

```
A>TYPE DIR88-07
EXE2BIN  EXE     2764  88-07-13    0:00
FILECONV EXE    32432  88-07-13    0:00
FIND     EXE     6573  88-07-13    0:00
SORT     EXE     2138  88-07-13    0:00
```

作成されたファイルの内容をタイプアウトして確認する

このような内容のファイル「DIR88-07」が作成されている。88年7月に作成されたファイルのみがピックアップされている

```
A>DIR DIR*
```

ここでの実行例で作成されたファイルをあらためて確認する

```
ドライブ A: のディスクのボリュームラベルはありません
ディレクトリは A:¥

DIR88-07           297  89-09-13   1:58
DIRSIZE            840  89-09-13   1:56
DIRNAME            840  89-09-13   1:57
     4 個のファイルがあります.
  868352 バイトが使用可能です.

A>
```

実習でこれらのファイルを作成した

処理を受けた順

図6.24c　ファイル内の文字列を検索して結果をファイルにセーブする

■ プリントアウトに関するコマンド ■

テキストファイル(文書ファイル)やコマンドの実行結果をプリントアウトする

［内部コマンド　　TYPE、COPY、リダイレクト］

A＞TYPE_*x*：パス名¥ファイル名_＞_PRN ⏎

ドライブ*x*：の「パス名」で指定されたディレクトリ上のテキストファイル(文字ファイル)「ファイル名」をプリントアウトする。TYPE コマンドの結果が、リダイレクトによりプリンタに出力される(このことは他のコマンドの実行結果のプリントアウトにも応用できる)。

A＞COPY_*x*：パス名¥ファイル名_PRN ⏎

上と同じファイルを、COPY コマンドで PRN(プリンタ)にコピー(プリントアウト)する。

A＞TYPE_*x*：パス名¥ファイル名⏎

［CTRL］＋［P］の入力後、TYPE コマンドを実行する。上と同じファイルが、コンソールに表示されるのと同時にプリントアウトされる(このことは他のコマンドの実行結果のプリントアウトにも応用できる)。［CTRL］＋［P］によるコンソールとプリンタの出力連動を解除するには、再度［CTRL］＋［P］を入力する。

［解説］　いずれもよく使われるコマンドであるが、リダイレクト(＞)の場合は『コマンドの実行結果』、［CTRL］＋［P］の場合は『画面の文字表示のすべて』がプリントアウトされる。COPY コマンドの場合は、コピー先がプリンタになる。これらのことに注意してそれぞれを使い分けるとよい。ただしいずれの場合も、PC-9800 シリーズの MS-DOS バージョン 3.*x* では、プリンタのデバイスドライバ「PRINT.SYS」を MS-DOS に組み込んでおく必要がある。

1) ここで［CTRL］+［P］をONにした場合
2) ここで［CTRL］+［P］をONにした場合

```
A>⏎
A>TYPE B:README.DOC⏎

1.    バックアップ作成時の注意


   ＰＳ９８－０１３（０１４）－ＨＵ／Ｈ４Ｗのシステムディスクは、９セクタ
 フォーマット（７２０ＫＢ）でフォーマットされています。
 ＰＳ９８－０１３（０１４）－ＨＵ／Ｈ４Ｗの
                          ∫
```

1) の場合は、このコマンドラインを含めてプリントアウトされる
2) の場合は、コマンドラインはプリントアウトされず、本文からプリントアウトされる

ドライブB：上のファイル「README.DOC」が画面表示と同時にプリントアウトされている

図 6.25　［CTRL］＋［P］の機能を利用してファイルをプリントアウトする

複数のファイルを一度のコマンドで順次プリントアウトする
ほかの仕事を行いながらファイルをプリントアウトする

[外部プログラム　PRINT.EXE　:　ファイル名にワイルドカードの使用可能]

A>PRINT_x：ファイル名 1_y：ファイル名 2_……⏎

ドライブx：、y：、…のカレントディレクトリ上のファイル「ファイル名 1」、「ファイル名 2」、…を連続してプリントアウトする(MS-DOS のバージョン 2.x では、カレントディレクトリ上のファイルに対してのみ有効だが、バージョン 3.x では、パス名を指定することによってカレントディレクトリ以外のファイルもプリントアウト可能)。プリントアウト中にほかのコマンドやプログラムの実行が可能である。

A>PRINT ⏎

プリントアウト開始後、プリントアウトが予約されているファイル名を表示する。

A>PRINT/T ⏎

プリントアウト開始後、ファイルのプリントアウトを中止する。ただし、プリンタへ送られてしまったデータはキャンセルできない。

[解説]　このPRINT コマンドは、プリントスプーラと呼ばれるもので、プリントアウトしながら、同時にほかのコマンドやプログラムの実行が可能である。つまり、バックグラウンドジョブ(裏仕事)としてプリントアウトが行われる。一度に最大 10 個のファイルのプリントアウトを指定できる(ファイル数はスイッチにより任意に設定可能であるがここでは省略する)。

```
A>PRINT CONFIG.SYS B:README.DOC⏎
出力装置を入力してください．[PRN]:⏎
PRINTコマンドの常駐部が組み込まれました．

    A:¥CONFIG.SYS を現在印刷中です．
    B:¥README.DOC は印刷を待っています．

A>
```

カレントドライブ(A：)上のファイル「CONFIG.SYS」と、ドライブB：上のファイル「README.DOC」をプリントアウトする

プリンタに出力する場合は⏎のみを入力すればよい

プリントアウトの状況が表示される

プリントアウトが開始されたが、プロンプトが表示されている。バックグラウンドでプリントアウトしながら、各種のコマンドやプログラムの実行が可能である

図 6.26a　バックグラウンドで複数ファイルのプリントアウトを行う

改ページが行われている

```
buffers = 10
files = 10
device = vjeb.drv /m1 /hb /j /l /ks /kr
```

1つのファイルのプリントアウトが終わると
フォームフィード（改ページ）が行われ、次
のファイルのプリントアウトが始まる

1番目のファイル「CONFIG.SYS」の紙面

1．　バックアップ作成時の注意

　ＰＳ９８－０１３（０１４）－ＨＵ／Ｈ４Ｗのシステムディスクは、９セクタフォーマット（７２０ＫＢ）でフォーマットされています。
ＰＳ９８－０１３（０１４）－ＨＵ／Ｈ４Ｗのバックアップを作成する時には、バックアップ媒体のフォーマット時に／９スイッチを指定してください。

（例）

　　Ａ＞ＦＯＲＭＡＴ　Ｂ：　／９

　また、ＭＥＮＵコマンドのメニュー項目「フロッピーディスク（６４０KBﾀｲﾌﾟ）のバックアップ」を選択することにより、７２０ＫＢフロッピーディスクの初期化からコピーまで続けて行うことができます。

　メニュー項目「フロッピーディスクのバックアップ（ドライブ指定）」は、ＰＣ－９８０１ＶＸなどのノーマルモード機種では、メニュー画面の２ページ目にあります。
　メニュー画面のページを切り替えるには、ROLLUPキー（次のページを表示），ROLLDOWNキー（前のページへ戻る）を押します。

2．　バックアップ作成時のドライブ指定

　ＭＥＮＵコマンドのメニュー項目「フロッピーディスクのバックアップ」と「フロッピーディスク（６４０KBﾀｲﾌﾟ）のバックアップ」は、コピー元，コピー先のドライブがそれぞれ A:，B: 固定になっています。
A:，B:以外のドライブでバックアップを行う場合には，メニュー項目「フロッピーディスクのバックアップ（ドライブ指定）」を選択してください。

　メニュー項目「フロッピーディスクのバックアップ（ドライブ指定）」は、メニュー画面の２ページ目にあります。　メニュー画面のページを切り替えるには、ROLL-UPキー（次のページを表示），ROLLDOWNキー（前のページへ戻る）を押します。

3．固定ディスク使用時の注意

　3．1．固定ディスクのフォーマット

　ＰＳ９８－０１３（０１４）では、どの機種でも固定ディスクを、拡張フォーマットで初期化することができます。
　しかし、拡張フォーマットできるのはＮＥＣ製の固定ディスクインターフェイスボード（ＰＣ－９８０１－０７／２７）を使用している場合のみです。

　また、ＰＣ－９８０１Ｆ３，Ｍ３，ＶＭ４や、ＰＣ－９８０１－０７／２７を実装して固定ディスクを接続している機種で、ＭＳ－ＤＯＳで固定ディスクを拡張フォーマットすると、ＭＳ－ＤＯＳの以前のバージョン（２．０／３．１）や、他のＯＳ（Ｎ88－日本語ＢＡＳＩＣやＣＰ／Ｍ－８６など）ではその固定ディスクを使用できなくなります。　他のＯＳ（ＭＳ　ＯＳ／２を除く）と固定ディスクを共有するような場合は、拡張フォーマットしないでください。

　3．2．固定ディスク起動メニュー

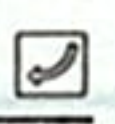
2番目のファイル「README.DOC」の紙面

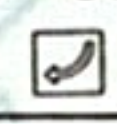
図 6.26b　プリントアウトの結果

■ RS-232C インターフェイスの入出力に関するコマンド ■

RS-232C インターフェイスを介してファイルを入出力する

［外部プログラム　　(COPYA.COM)（注：機種によりコマンド名が異なる）］

A＞COPYA_*x*：パス名¥ファイル名_AUX↵

ドライブ*x*：の「パス名」で指定されたディレクトリ上のファイル「ファイル名」をRS-232C インターフェイスから出力する。

A＞COPYA_AUX_*x*：パス名¥ファイル名↵

RS-232C インターフェイスから入力されたデータを、ドライブ*x*：の「パス名」で指定されたディレクトリ上のファイルに「ファイル名」としてセーブする。

［解説］　RS-232C インターフェイスは、通常、デバイスファイル名「AUX」に割り当てられている。本来、この実行には、周辺装置間のファイル転送プログラムである COPY コマンドが使えるはずであるが、正常に動作しない場合があるため、各社独自の専用コマンドを設けている場合が多い。ここでは、PC-9800 シリーズにおける専用コマンド「COPYA.COM」（外部プログラム）の例を示す。このコマンドは、テキストファイル（文字ファイル）に対してのみ入出力が可能である。なお、RS-232C インターフェイスは、使用する前に一度、SPEED コマンドで初期設定する必要がある（一度行えば電源を OFF するまでは有効）。また、PC-9800 シリーズの MS-DOS のバージョン 3.*x* では、デバイスドライバ「RSDRV.SYS」を MS-DOS に組み込んでおかなければ、COPYA コマンドは機能しない。

PC-9801シリーズのMS-DOSの専用プログラム

A>COPYA README.DOC AUX ↵ ……カレントドライブ(A：)上のファイル「README.DOC」を RS-232Cインターフェイスから出力する

ファイル転送を終了しました

A>COPYA AUX RECIEVE ↵ ……RS-232Cインターフェイスから入力したデータを、ドライブB：上に「RECEIVE」というファイル名でセーブする

ファイル転送を終了しました

A>

（RS-232Cインターフェイスで通信する場合は、互いに、ボーレート（転送速度）やストップビット等の設定を合わせておくこと）

図 6.27　RC-232C インターフェイスによるファイルの入出力

■ 階層ディレクトリに関するコマンド ■

任意のディレクトリ上のユーザープログラムを実行する

［解説］　ユーザープログラム(外部プログラム)は、MS-DOSのバージョン2.*x*の場合、カレントディレクトリ上のもの、および後述の「PATH」コマンドにより、あらかじめ設定したディレクトリ上のもののみ実行可能である。

またMS-DOSのバージョン3.*x*以降の場合、パス名を指定することにより、カレントディレクトリに関係なく、任意のディレクトリ上のプログラムを直接実行できる。なおこれらの実行例は、このあとのそれぞれの項目や、319ページの図6.33で行う。

カレントディレクトリをチェンジする

［内部コマンド　　CD(CHDIR)］

A>CD_x：パス名⏎

ドライブ*x*：上のカレントディレクトリを「パス名」で指定されたディレクトリにチェンジする。

A>CD_x：..⏎

ドライブ*x*：上のカレントディレクトリを、現在のディレクトリの親ディレクトリにチェンジする。「..」は親ディレクトリの代用記号である(なお「.」は、カレントディレクトリ自身の代用記号である)。

A>CD_x：⏎

ドライブ*x*：における現在のカレントディレクトリを表示する。

［解説］　コマンドは、「CD」または「CHDIR」のどちらでも実行可能。さまざまな作業を行うにあたり、カレントディレクトリをどのディレクトリにしておくかは、作業能率に大きく関係する。それぞれの場合に応じて、もっとも都合のよいディレクトリにカレントディレクトリをチェンジしておくのがよい。

```
A>CD B:⏎ ……………ドライブB：のカレントディレクトリを見る
B:¥ ……………ドライブB：のカレントディレクトリはルートディレクトリ

A>B:EDLIN B:AUTOEXEC.BAT ……ドライブB：のカレントディレクトリ(ルートディレクトリ)上のEDLINを実行する
コマンドまたはファイル名が違います. ……………しかしルートディレクトリにはプログラム「EDLIN.EXE」が
                                                ないので実行できない

A>B:¥MSDOS¥TOOL¥EDLIN B:AUTOEXEC.BAT⏎ ……………ドライブB：のディレクトリ[TOOL]のなかのEDLIN
                                                 をパスを指定して実行する
新しいファイルです.
*I⏎
    1:*SET PATH=A:¥;B:¥⏎        MS-DOS Ver3.x以降ではパスを指定すれば
    2:*^C                        目的のコマンドを実行できる。
                                 EDLINで自動スタート・バッチファイルを作成した
*E⏎

A>DIR B:AUTOEXEC.BAT⏎ ……………ドライブB：のカレントディレクトリ(ルートディレクトリ)上の
                                 ファイル「AUTOEXEC.BAT」を確認する

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥
                  ルートディレクトリ上に作成されたファイル「AUTOEXEC.BAT」
AUTOEXEC BAT          19  89-09-01   21:22
         1 個のファイルがあります.
   748544 バイトが使用可能です.

A>CD B:¥MSDOS¥TOOL⏎ ……………ドライブB：のカレントディレクトリを[TOOL]にチェンジする

A>CD B:⏎
B:¥MSDOS¥TOOL ……………ドライブB：のカレントディレクトリを確認する。
                      ドライブB：のカレントディレクトリは[TOOL]

A>B:EDLIN B:AUTOEXEC.BAT⏎ ……………最初のコマンドを再度実行
新しいファイルです.
*I⏎
    1:*SET PATH=B:¥MSDOS¥TOOL⏎     カレントディレクトリに目的のプログラムが
    2:*^C                          存在すれば実行される。
                                   今回は、ファイルが作成されるディレクトリが
                                   前回と異なる
*E⏎

A>DIR B:AUTOEXEC.BAT⏎ ……………ドライブB：のカレントディレクトリ(ディレクトリ[TOOL])上の
                                 ファイル「AUTOEXEC.BAT」を確認する

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥MSDOS¥TOOL ←カレントディレクトリは[TOOL]

AUTOEXEC BAT          25  89-09-01   21:23
         1 個のファイルがあります.
   747520 バイトが使用可能です.
  ディレクトリ[TOOL]上に作成されたファイル「AUTOEXEC.BAT」
A>
```

```
                                                       ドライブB：の階層ディレクトリの状態
ル ├─ MSDOS ─┬─ CHKDSK.EXE
ー │          │             ┌─ EDLIN.EXE
ト │          └─ TOOL ─────┴─ AUTOEXEC.BAT
   └─ AUTOEXEC.BAT                ↑
            ↑                     └── ディレクトリ[TOOL]の下に作成された
            │                         ファイル「AUTOEXEC.BAT」
            └── ルートディレクトリの下に作成されたファイル「AUTOEXEC.BAT」
```

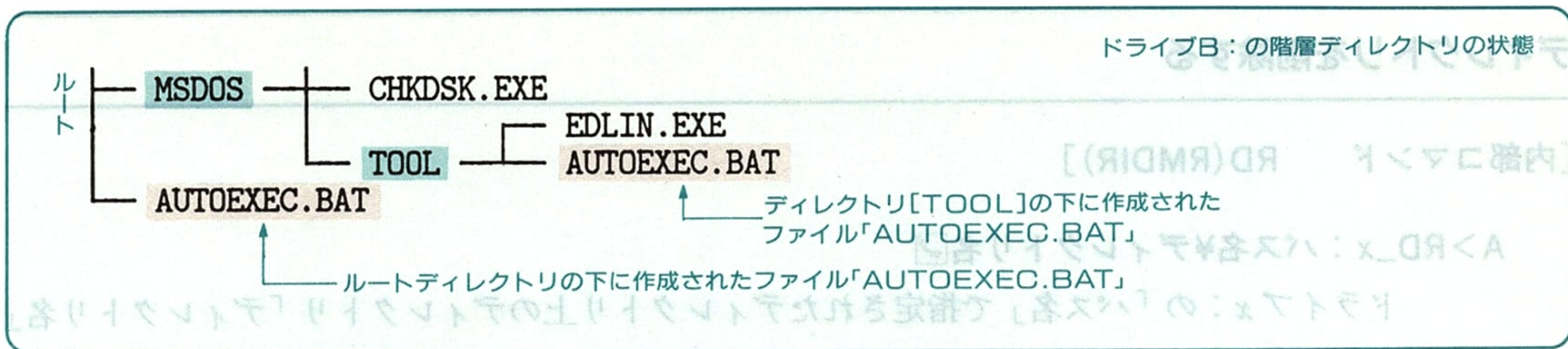



図 6.28 カレントディレクトリをチェンジする

ディレクトリを作成する

［内部コマンド　　MD（MKDIR）］

A＞MD_*x*：パス名¥ディレクトリ名⏎

ドライブ*x*：の「パス名」で指定されたディレクトリ上に、新しいディレクトリ「ディレクトリ名」を作成する。

［解説］　コマンドは、「MD」または「MKDIR」のどちらでも実行可能。ディレクトリが異なれば、同じディレクトリ名がいくつ存在してもよい。ディレクトリを1つ作成すると、1Mバイトタイプのフロッピーディスクでは1Kバイト、ハードディスクでは数Kバイト（186ページの4.4章を参照）を消費する。

```
B>MD ¥ETC⏎ ……………カレントドライブ(B：)のルートディレクトリ[¥]上に、
                   サブディレクトリ[ETC]を作成する
B>MD ¥MSDOS¥ETC⏎ ……………カレントドライブ(B：)のディレクトリ[MSDOS]上に、
                          サブディレクトリ[ETC]を作成する
B>MD ETC⏎ ……………カレントドライブ(B：)のカレントディレクトリ(先の実行例で[TOOL]に
                  移ったまま)上に、ディレクトリ[ETC]を作成する
     ↑
B>   ¥がないことに注目
```

```
                                              ドライブB：の階層ディレクトリの状態
ル ├─ MSDOS ─┬────────────── ┌─ EDLIN.EXE
ー │          ├─ TOOL ───────┴─ ETC ─
ト │          │    ↑
   │          │    └──── カレントディレクトリ
   │          └─ ETC ─
   │                  3つの同名のディレクトリ[ETC]が作成されている。
   └─ ETC ─           ディレクトリが異なれば、同名のディレクトリやファイル
                      が存在してもよい
```

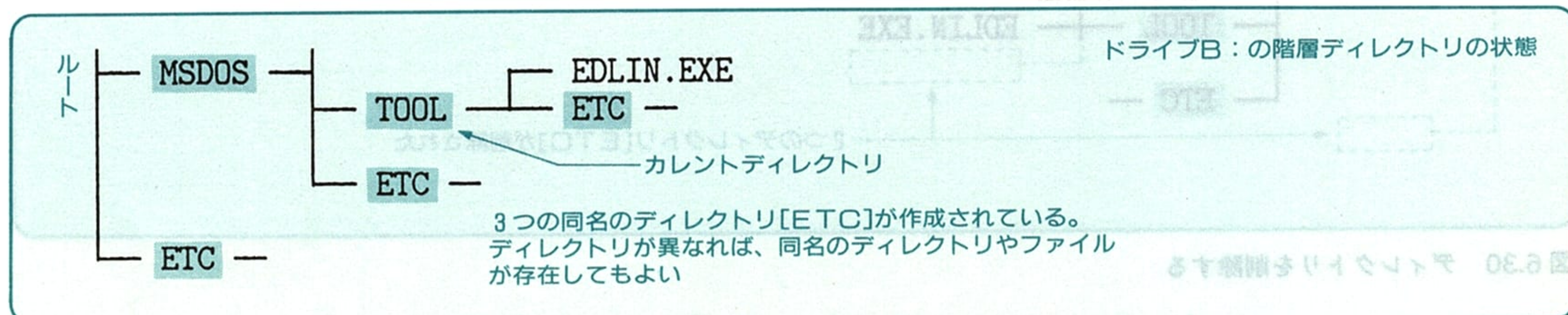
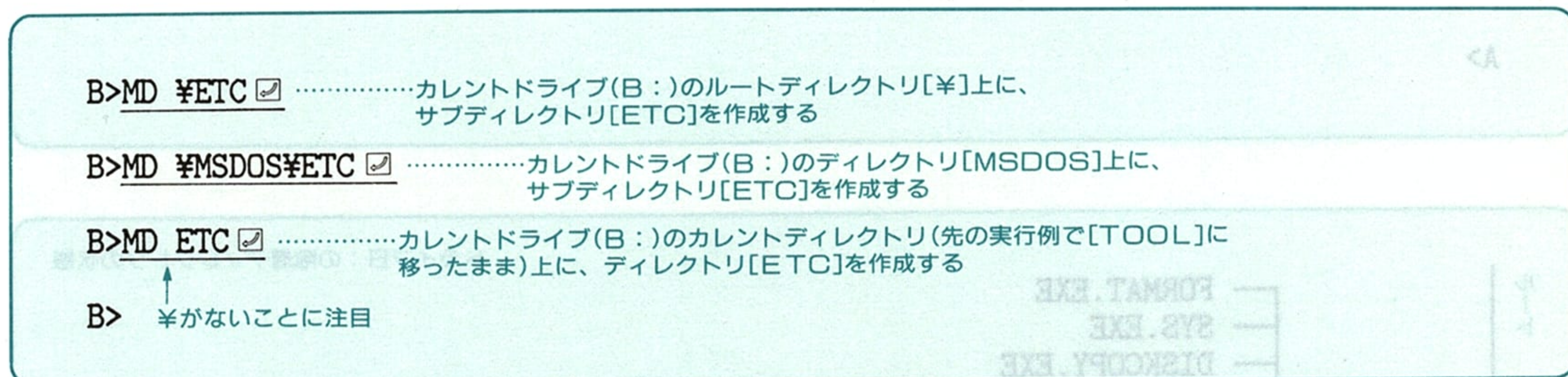



図 6.29 新しいディレクトリを作成する

ディレクトリを削除する

［内部コマンド　　RD(RMDIR)］

A＞RD_*x*：パス名¥ディレクトリ名⏎

ドライブ*x*：の「パス名」で指定されたディレクトリ上のディレクトリ「ディレクトリ名」を削除する。

［解説］　コマンドは、「RD」または「RMDIR」のどちらでも実行可能。削除しようとするディレクトリがカレントディレクトリの場合や、ファイルあるいはサブディレクトリが存在している場合は削除ができない。つまり、まったくの空ディレクトリで、かつカレントディレクトリでない場合のみ削除可能。

```
A>RD B:¥ETC⏎ ……………ドライブB：のルートディレクトリ上のディレクトリ[ETC]を削除する

A>RD B:¥ETC⏎ ……………ルートディレクトリ上のディレクトリ[ETC]は、すでに削除されて存在していない
パスの指定が違うかディレクトリでないか
またはディレクトリが空ではありません.

A>RD B:ETC⏎ ……………カレントディレクトリ(現在[TOOL]にいる)上のディレクトリ[ETC]を削除する

A>RD B:¥MSDOS⏎ ……………ドライブB：のルートディレクトリ上のディレクトリ[MSDOS]を削除する
パスの指定が違うかディレクトリでないか ……………空のディレクトリではないので削除はできない
またはディレクトリが空ではありません.

A>
```

```
                                              ドライブB：の階層ディレクトリの状態
ル │          ┌── FORMAT.EXE
ー │          ├── SYS.EXE
ト │          ├── DISKCOPY.EXE
   └── MSDOS ─┼── CHKDSK.EXE
              ├── TOOL ──┬── EDLIN.EXE
              │          └ [     ]
              └── ETC ─
   └ [     ] ←──────────── 2つのディレクトリ[ETC]が削除された
```

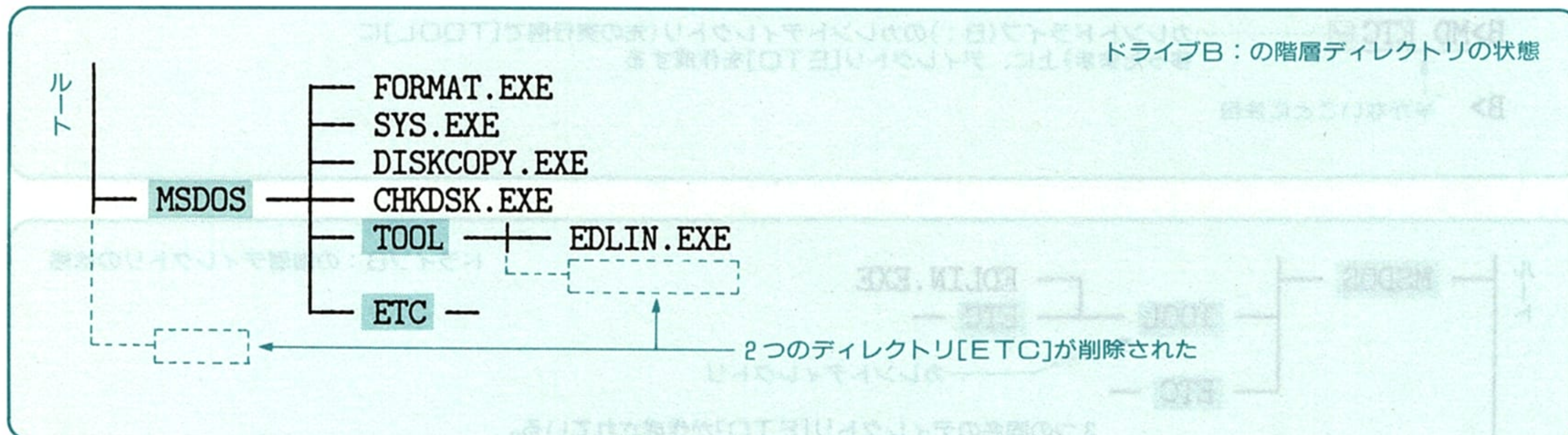



図 6.30　ディレクトリを削除する

ディレクトリ名を変更する

[<3.*x*>外部コマンド　　RENDIR.COM]

A>RENDIR_*x*：パス名¥ディレクトリ名 1_ディレクトリ名 2 ↵

ドライブ *x*：の「パス名」で指定されたディレクトリ上のディレクトリ「ディレクトリ名 1」を、「ディレクトリ名 2」に変更する。

[解説]　必要性が高いのに、MS-DOS バージョン 2.*x* には用意されていなかったディレクトリ名を変更するためのコマンドである。ファイル名の変更には、REN コマンドを用いる。

```
A>DIR ↵ ……………カレントドライブ(A:)上のディレクトリを見る

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

COMMAND  COM     24931  88-07-13   0:00
CONFIG   SYS        87  89-07-29  17:25
PRINT    SYS      5855  88-07-13   0:00
WP           <DIR>      89-07-30  15:21
MP           <DIR>      89-07-30  15:21
MSDOS        <DIR>      89-07-30  15:21
        6 個のファイルがあります.
   749568 バイトが使用可能です.

A>RENDIR ¥MSDOS MSDOS33 ↵ ……………ディレクトリ[MSDOS]を[MSDOS33]にリネームする
                         (カレントディレクトリがルートなので、この場合「¥」は省略できるが)

A>DIR ↵ ……………実行後のディレクトリを確認する

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

COMMAND  COM     24931  88-07-13   0:00
CONFIG   SYS        87  89-07-29  17:25
PRINT    SYS      5855  88-07-13   0:00
WP           <DIR>      89-07-30  15:21
MP           <DIR>      89-07-30  15:21
MSDOS33      <DIR>      89-07-30  15:21
        6 個のファイルがあります.
   749568 バイトが使用可能です.
  └───ディレクトリ名が変更されている
A>
```

図 6.31　ディレクトリ名を変更する

階層ディレクトリの状況を表示する

[<3.*x*>外部コマンド　　TREE.EXE]

A>TREE_*x*：⏎
　ドライブ*x*：上のすべてのディレクトリを、パス名とともに表示する。

A>TREE_*x*：/F⏎
　上の例で、ディレクトリだけでなく、そのディレクトリに含まれるファイル名の一覧も表示する。

[解説]　DIR コマンドは、指定したディレクトリ上に直接存在するファイルおよびサブディレクトリしか表示できないのに対して、TREE コマンドはディスク上のすべてのディレクトリと、そのすべてのファイル名を表示することができる。また、これと同様のことは「CHKDSK /V」コマンドでも可能。

```
A>TREE A:⏎  ……………ドライブA：の全ディレクトリを表示する

ディレクトリ・パス・リスト

パス: A:¥WP
                                  } パス名とサブディレクトリの有無が表示される
サブディレクトリ: BUN
                  JXW


パス: A:¥WP¥BUN

サブディレクトリ: なし
     ⋮
A>TREE A: /F⏎  ……………ドライブA：の全ディレクトリと、そのすべてのファイルを表示する

ディレクトリ・パス・リスト
                    ◄……………ルートディレクトリの場合、「パス」の表示がない
ファイル:           COMMAND .COM
                    CONFIG  .SYS  } ルートディレクトリ上に含まれるファイル
                    PRINT   .SYS
     ⋮
```

```
      :
      :
ファイル:          NMSD1
                   NMSD2
                   EIGY05  .BUN
                   EIGY06  .BUN
                   EIGY07  .BUN


パス: A:¥WP¥JXW

サブディレクトリ: なし

ファイル:          なし


パス: A:¥MP

サブディレクトリ: なし

ファイル:          URIAGE  .MP
                   MACHINE .MP
                   MITUMORI.MP


パス: A:¥MSDOS

サブディレクトリ: TOOL

ファイル:          FORMAT  .EXE
                   SYS     .EXE
                   DISKCOPY.EXE
                   CHKDSK  .EXE


パス: A:¥MSDOS¥TOOL

サブディレクトリ: なし

ファイル:          EDLIN   .EXE


A>
```

このようにサブディレクトリに含まれるファイルも表示される

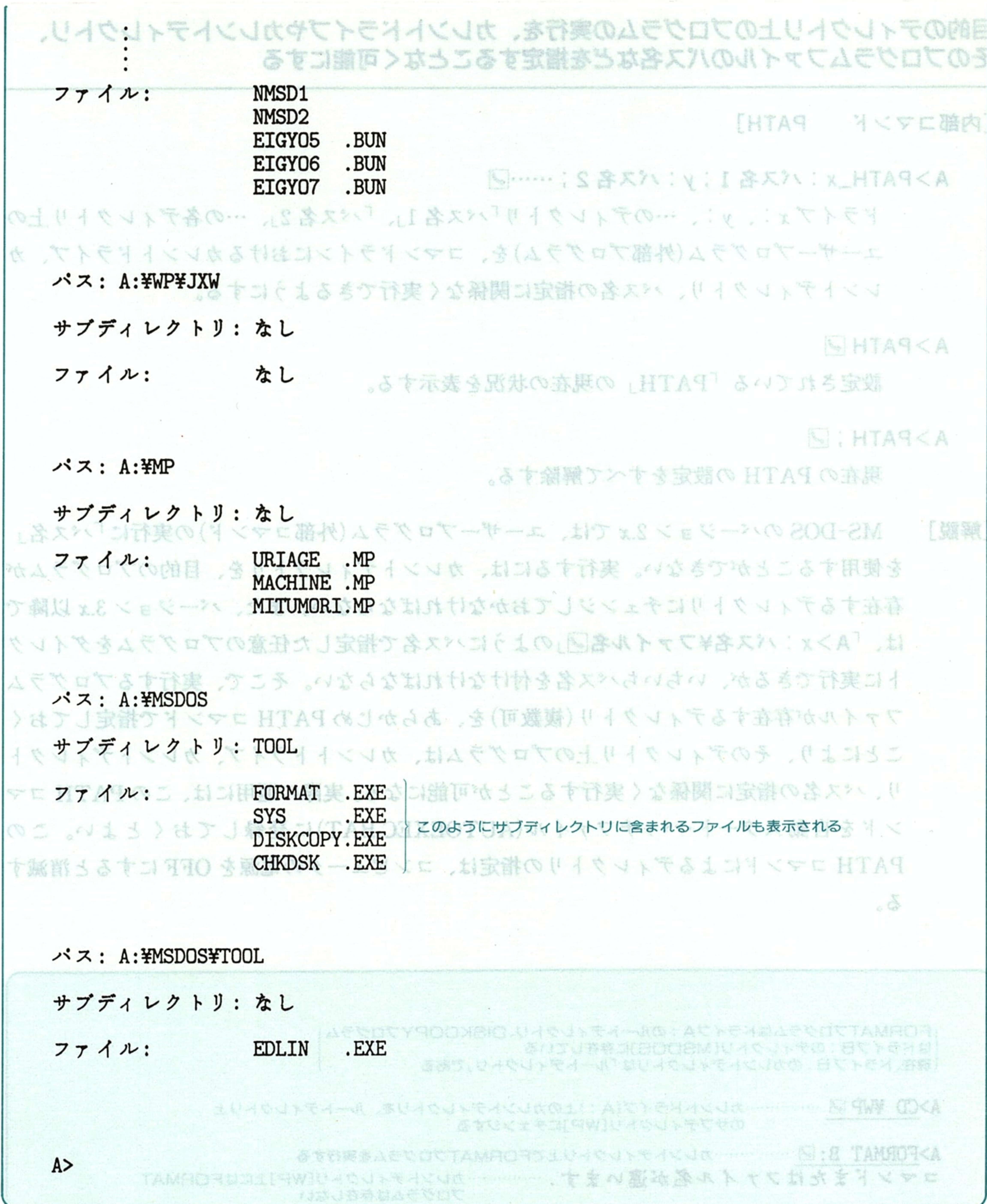



図 6.32　ディスク上の全ディレクトリと全ファイルの表示

目的のディレクトリ上のプログラムの実行を、カレントドライブやカレントディレクトリ、そのプログラムファイルのパス名などを指定することなく可能にする

［内部コマンド　　PATH］

A＞PATH_*x*：パス名 1；*y*：パス名 2；……⏎

ドライブ*x*：、*y*：、…のディレクトリ「パス名 1」、「パス名 2」、…の各ディレクトリ上のユーザープログラム(外部プログラム)を、コマンドラインにおけるカレントドライブ、カレントディレクトリ、パス名の指定に関係なく実行できるようにする。

A＞PATH⏎

設定されている「PATH」の現在の状況を表示する。

A＞PATH；⏎

現在の PATH の設定をすべて解除する。

［解説］　MS-DOS のバージョン 2.*x* では、ユーザープログラム(外部コマンド)の実行に「パス名」を使用することができない。実行するには、カレントディレクトリを、目的のプログラムが存在するディレクトリにチェンジしておかなければならない。また、バージョン 3.*x* 以降では、「**A＞*x*：パス名¥ファイル名⏎**」のようにパス名で指定した任意のプログラムをダイレクトに実行できるが、いちいちパス名を付けなければならない。そこで、実行するプログラムファイルが存在するディレクトリ(複数可)を、あらかじめ PATH コマンドで指定しておくことにより、そのディレクトリ上のプログラムは、カレントドライブ、カレントディレクトリ、パス名の指定に関係なく実行することが可能になる。実際の運用には、この PATH コマンドを自動スタート・バッチファイル(AUTOEXEC.BAT)に登録しておくとよい。このPATH コマンドによるディレクトリの指定は、コンピュータの電源を OFF にすると消滅する。

（FORMATプログラムはドライブA：のルートディレクトリ、DISKCOPYプログラムはドライブB：のディレクトリ[MSDOS]に存在している
現在、ドライブB：のカレントディレクトリは「ルートディレクトリ」である）

A>CD ¥WP⏎ ……………カレントドライブ(A：)上のカレントディレクトリを、ルートディレクトリ上のサブディレクトリ[WP]にチェンジする

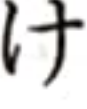
A>FORMAT B:⏎ ……………カレントディレクトリ上でFORMATプログラムを実行する
コマンドまたはファイル名が違います. ……………カレントディレクトリ[WP]上にはFORMATプログラムは存在しない

```
A>B:DISKCOPY B: A:⏎ ……ドライブB：のカレントディレクトリ(ルートディレクトリ)
                        上でDISKCOPYプログラムを実行する
コマンドまたはファイル名が違います. ……カレントディレクトリ[¥(ルートディレクトリ)]
                        にはDISKCOPYプログラムは存在しない

A>PATH A:¥;B:¥MSDOS⏎ ……ドライブA：のルートディレクトリ、およびドライブB：の
                        [MSDOS]にPATHを設定する

A>FORMAT B:⏎ ……カレントディレクトリ[WP]上で、ルートディレクトリの
Format Version 4.10     FORMATプログラムを実行する

                        実行された
新しいディスクをドライブ B: に挿入し
どれかキーを押してください^C ……CTRL+C を入力して当プログラムを中止して
                        MS-DOSにもどる

A>DISKCOPY B: A:⏎ ……カレントドライブ(A：)のディレクトリ[WP]上で、ドライブB：のディレクトリ
 └「B：」がないことに注目  [MSDOS]上のプログラム[DISKCOPY]を実行する
DISKCOPY version 4.00

ディスクのコピーを行います       カレントドライブがA：であるのに「B：」の指定なしにB：上の
                        カレントディレクトリでないプログラムが実行された

送り側ディスクをドライブ B: に挿入してください
受け側ディスクをドライブ A: に挿入してください
準備ができたらどれかのキーを押してください^C ……MS-DOSにもどる

A>
```

図 6.33a 「PATH」を設定して、任意のディレクトリのユーザープログラムの実行を可能にする

```
A>B:DISKCOPY B: A:⏎ ……ドライブB：のカレントディレクトリ(ルートディレクトリ)上で
                        DISKCOPYプログラムを実行する
コマンドまたはファイル名が違います. ……カレントディレクトリ(ルートディレクトリ)には
 パス名を含めてファイルを指定          DISKCOPYプログラムは存在しない
A>B:¥MSDOS¥DISKCOPY B: A:⏎ ……MS-DOSのバージョン3.x以上ではパス名を指定することに
                        より任意のディレクトリのプログラムを実行することができる
DISKCOPY version 4.00

ディスクのコピーを行います

送り側ディスクをドライブ B: に挿入してください
受け側ディスクをドライブ A: に挿入してください
準備ができたらどれかのキーを押してください^C ……MS-DOSにもどる

A>
```

図 6.33b MS-DOS のバージョン 3.*x* 以上では、パス名で指定した任意のプログラムを直接実行できる
(「PATH」の設定とは関係ない)

階層構造ごと任意のディレクトリ上のすべてのファイルをコピーする

［＜3.*x*＞外部コマンド　　XCOPY.EXE　：　ファイル名にワイルドカードの使用可能］

A＞XCOPY_*x*：パス名1_*y*：パス名2_/S↵

「/S」スイッチにより、ドライブ*x*：の「パス名1」で指定されたディレクトリ上の、すべてのサブディレクトリとそのすべてのファイルを、階層ごとそっくりドライブ*y*：上の「パス名2」で指定されたディレクトリ上にコピーする(ただし、ファイルやサブディレクトリが存在しない空ディレクトリはコピー(作成)されない)。

「パス名2」で指定されたコピー先のディレクトリ名が存在しない場合、XCOPYコマンドの問い合せにしたがって答えると、そのディレクトリが作成された上で、コピーが行われる。

A＞XCOPY_*x*：パス名1_*y*：パス名2_/S/E↵

さきの「/S」スイッチ付きのコマンドに、さらに「/E」スイッチをつけることにより、空ディレクトリもコピー(作成)することができる。なお各スイッチは、↵の直前に付ける。

［解説］　まず最初に、このコマンドはMS-DOSのコマンドとしては作りが少々変則的であり、「XCOPY.EXE」を「MCOPY.EXE」にリネームし、「MCOPYコマンド」として利用することを勧める(ここでの呼び方はいちおう、XCOPYとしておくが)。MCOPYコマンドの場合は、コマンドラインのコピー元やコピー先の指定が、ファイルであるのか、ディレクトリであるのかを自動的に判別し、それにしたがった正しい処理を行ってくれる(図6-34aのような問い合せは発生しない)。

XCOPYコマンドは、「/S」スイッチを付けない場合、基本的にはCOPYコマンドと同じような働きをするが、「/S」スイッチを付けることにより、指定したディレクトリに連なる階層構造のすべてを、そっくりコピーすることができる。たとえば「パス名1」にルートディレクトリを指定すれば、そのディスク上のすべての階層構造をそっくりコピーできることになる。ただし、DISKCOPYコマンドとはコピー方法が本質的に異なり、あくまでCOPYコマンドと同様のファイル単位のコピーである。

なお、XCOPYコマンドには、その他のスイッチの機能により、ファイルのアーカイブ属性や、作成日付などにより、条件選択コピーを行うことができる(これについては「ディスクファイルに関するコマンド」におけるXCOPYコマンドの項(287ページ)を参照)。なお、当コマンドは、4.5章のXCOPYコマンドの項(205ページ)でくわしく解説している。

```
              ┌──この「A：」は省略可
A>XCOPY A:¥WP B:¥WP /S/E↵ ……ドライブA：のディレクトリ[WP]上のすべてのサブディレクトリ(その下にファイルが存在しないものも含めて)とファイルを階層ごとドライブB：のディレクトリ[WP]に上コピーする

WP は受け側のファイル名ですか．それともディレクトリ名ですか．  ドライブB：に「WP」が存在しない場合、問い合せがある
<F = ファイル名, D = ディレクトリ名>? D
                                        └──ディレクトリを指定するとドライブB：にディレクトリ[WP]が作成される
送り側のファイルを読込み中です．．．
A:¥WP¥BUN¥NMSD1
A:¥WP¥BUN¥NMSD2
A:¥WP¥BUN¥EIGY05.BUN
A:¥WP¥BUN¥EIGY06.BUN
A:¥WP¥BUN¥EIGY07.BUN
        5 個のファイルをコピーしました．

A>TREE B: /F↵ ……ドライブB：のディレクトリの階層構造を確認してみる

ディレクトリ・パス・リスト
ファイル:           なし

パス: B:¥WP ……ディレクトリ[WP]が作成されている
サブディレクトリ: BUN
                  JXW
ファイル:           なし

パス: B:¥WP¥BUN
サブディレクトリ: なし
ファイル:           NMSD1
                  NMSD2
                  EIGY05   .BUN
                  EIGY06   .BUN
                  EIGY07   .BUN

パス: B:¥WP¥JXW
サブディレクトリ: なし
ファイル:           なし
  (サブディレクトリやファイルが存在しないディレクトリもコピーされている)
  (ドライブA：のディレクトリ[WP]上のすべてが階層ごとコピーされている)

A>
```

図 6.34a　サブディレクトリ上のファイルを階層構造ごとそっくりコピーする

```
A>DIR
 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

COMMAND  COM     24931  88-07-13    0:00
CONFIG   SYS        87  89-07-29   17:25
PRINT    SYS      5855  88-07-13    0:00
MP           <DIR>      89-09-14   21:26
WP           <DIR>      89-09-14   21:26
MSDOS        <DIR>      89-09-14   21:26
        6 個のファイルがあります.
   750592 バイトが使用可能です.

A>DIR B:
 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

ファイルが見つかりません.

A>XCOPY A:¥ B: /S
送り側のファイルを読込み中です. . .
A:COMMAND.COM
A:CONFIG.SYS
A:PRINT.SYS
A:MP¥URTAGE.MP
  ～～～～～～～～～～
...¥BUN¥NMSD2
A:WP¥BUN¥EIGYO5.BUN
A:WP¥BUN¥EIGYO6.BUN
A:WP¥BUN¥EIGYO7.BUN
        16 個のファイルをコピーしました.

A>DIR B:
 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

COMMAND  COM     24931  88-07-13    0:00
CONFIG   SYS        87  89-07-29   17:25
PRINT    SYS      5855  88-07-13    0:00
MP           <DIR>      89-09-14   21:57
WP           <DIR>      89-09-14   21:57
MSDOS        <DIR>      89-09-14   21:57
        6 個のファイルがあります.
   749568 バイトが使用可能です.

A>
```

- A>DIR ……ドライブA：のディレクトリを確認する
- A>DIR B: ……ドライブB：のディレクトリを確認する
- A>XCOPY A:¥ B: /S ……ドライブA：のディスク上のすべてのファイルを階層構造ごとそっくりドライブB：にコピーする。オプション「/S」と、コピー元のディレクトリが「¥」(ルート)であることに注目
  - この「A：」は省略可
- A>DIR B:（2回目）……コピー先のドライブB：のディレクトリを確認する
- COMMAND～MSDOSの各行：ドライブA：のディスク全体がそっくりコピーされている
- MP・WP・MSDOS：オプション「/S」をつけたのでサブディレクトリもコピーされている(もちろんそれぞれの中身もコピーされている)

図6.34b　ディスクの全ファイルを階層構造ごとそっくりコピーする

ファイルのバックアップコピーを行う

［<3.*x*>外部コマンド　BACKUP.EXE　：　ファイル名にワイルドカードの使用可能］

A>BACKUP_*x*：パス名_*y*：_/S↵

ドライブ*x*：の「パス名」で指定されたディレクトリ上の、すべてのサブディレクトリとそのすべてのファイルを、階層ごとそっくり(「/S」スイッチの機能)ドライブ*y*：上にバックアップコピーする。

その他の主要なスイッチの機能には、次のようなものがある。

/M ………前回のBACKUPコマンドの実行以降に作成または更新されたファイル(アーカイブ属性がついているファイル)のみをバックアップコピーする。

/A ………バックアップ先のディスク上のファイルを削除せず、そのディスクに追加コピーする。

/L：*z*：パス名¥ファイル名 ………バックアップ実行の日時や対象ファイル名一覧を記録した履歴ファイルを、指定したディレクトリ上のファイル「ファイル名」として作成する。「ファイル名」などを省略すると、バックアップ元のディスクのルートディレクトリに、「BACKUP.LOG」というファイルが自動的に作成される。

/D：89-09-01 ………89年9月1日以降に作成または更新されたファイルのみをバックアップコピーする。

/T：15：30 ……………15時30分以降に作成または更新されたファイルのみをバックアップコピーする。

［解説］　BACKUPコマンドは、普通、ハードディスクからフロッピーディスクへ、ファイルをバックアップコピーする際に利用される。とくに気をつける必要があるのは、「/A」を付けない場合、バックアップ先のフロッピーディスクの内容がすべて削除され、そのあとでコピーが行われる点である。いずれも、BACKUPコマンドでコピーされたファイルは、次項のRESTOREコマンドで復元しなければ使用することはできない。このほかにもいくつかスイッチがあるが、ここでは省略する。なお、BACKUPコマンドおよび次項のRESTOREコマンドについては、195ページの4.5章でくわしく解説している。

```
A>DIR⏎ ・・・・・・・・・・・・ドライブA：のディスクの内容を見る

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

COMMAND  COM    24931  88-07-13   0:00
CONFIG   SYS       87  89-07-29  17:25
PRINT    SYS     5855  88-07-13   0:00
MP            <DIR>    89-09-14  20:36
WP            <DIR>    89-09-14  20:37
MSDOS         <DIR>    89-09-14  20:36
        6 個のファイルがあります.
   750592 バイトが使用可能です.
          ┌──この「A：」は省略可
A>BACKUP A:¥ B: /S⏎ ・・・・・・・・・・ドライブA：の内容をドライブB：にバックアップする。オプション「/S」を
                                  をつけて、サブディレクトリの内容もすべてをバックアップする

送り側ディスクを挿入してください.
準備ができたら何かキーを押してください.⏎

バックアップディスク 01 をドライブ B: に挿入してください.
警告!
受け側ドライブ B: において, ルートディレクトリの下のファイルは消去されます.
準備ができたら何かキーを押してください.⏎ ・・・・・・・・・・ドライブB：の内容はすべて消去
                                              される。ここでCTRL+Cを入
*** ファイルをドライブ B: にバックアップします.***    力すれば、BACKUPコマンド
ディスク番号は:  01                                 をキャンセルしてMS-DOSに
                                              もどることができる
¥IO.SYS
¥MSDOS.SYS
¥COMMAND.COM
¥CONFIG.SYS
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
¥MSDOS¥SYS.EXE
¥MSDOS¥DISKCOPY.EXE
¥MSDOS¥CHKDSK.EXE
¥MSDOS¥TOOL¥EDLIN.EXE

A>DIR/W B:⏎ ・・・・・・・・・・・ドライブB：の内容を確認する

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥
 ┌──────バックアップの管理情報（サブディレクトリの構成など）を収めたファイルが作成される
BACKUPID @@@    COMMAND  COM    CONFIG   SYS◄┐  PRINT    SYS    URIAGE   MP
MACHINE  MP     MITUMORI MP     FORMAT   EXE │  SYS      EXE    DISKCOPY EXE
CHKDSK   EXE    NMSD1           NMSD2        │  EIGYO5   BUN    EIGYO6   BUN
EIGYO7   BUN    EDLIN    EXE             バックアップされたファイルは、ファイル形式が変わって
      17 個のファイルがあります.          しまうので、たとえばこのファイルなどをTYPEコマンド
    751616 バイトが使用可能です.          でタイプアウトしようとしても正しく表示されない

A>
```

図6.35　サブディレクトリを含めた全ファイルをバックアップする

バックアップコピーから事故ファイルを復元する

［＜3.*x*＞外部コマンド　　RESTORE.EXE］

A＞RESTORE_*x*：_*y*：パス名_/S↵

BACKUP コマンドでバックアップした、ドライブ*x*：のバックアップディスクから、ドライブ*y*：の「パス名」で指定したディレクトリ上に、サブディレクトリも含めて(「/S」スイッチの機能)ファイルを復元する。

その他の主要なスイッチの機能には、次のようなものがある。

/P ………ファイルが「リードオンリーファイル」もしくは「システムファイル」の場合、それらを復元してよいかどうかの問い合わせがある。

/M ………前回の BACKUP コマンドの実行以降に削除または更新されたファイル(アーカイブ属性がついているファイル)のみをバックアップファイルで復元する。

/N ………復元先のディスクに存在しないファイルだけを復元する。

/B：89-08-31
/A：89-09-01

これらの日付で指定した以前に作成されたファイル(「/B：」スイッチの場合)、もしくは以降に作成または更新されたファイル(「/A：」スイッチの場合)だけを復元する。なお、スイッチ「/E：15：29」(以前)、「/L：15：30」(以降)により、ファイルの作成時刻による同様の選択ができる。

［解説］　RESTORE コマンドは、BACKUP コマンドで作成されたバックアップディスクの内容を、ハードディスクに復元する場合に使用する。ただし、MS-DOS のシステムファイル(IO.SYS、MSDOS.SYS)を復元する場合、システムによっては起動ができなくなることがあるので注意を要する。その際には、「/P」スイッチを付け、復元する／しないをマニュアル選択し、これらのファイルを復元しないようにする(SYS コマンドなどで別途コピーする)。なお、RESTORE コマンドおよび前項の BACKUP コマンドについては、193 ページの 4.5 章でくわしく解説している。

```
A>DIR⏎ ……………ドライブA：のディスクの内容を見る

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

ファイルが見つかりません. ……………空のディスク

A>DIR/W B:⏎ ……………ドライブB：のバックアップしたディスク(図6.35で
                        作成したディスク)の内容を見る
 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

BACKUPID @@@     COMMAND  COM    CONFIG   SYS    PRINT    SYS    URIAGE   MP
MACHINE  MP      MITUMORI MP     FORMAT   EXE    SYS      EXE    DISKCOPY EXE
CHKDSK   EXE     NMSD1           NMSD2           EIGYO5   BUN    EIGYO6   BUN
EIGYO7   BUN     EDLIN    EXE
        17 個のファイルがあります.
   751616 バイトが使用可能です.

A>RESTORE B: A:¥ /S⏎ ……………ドライブB：のバックアップディスクを、ドライブA：の空ディスク上に
                        復元する。/Sオプションをつけて、サブディレクトリもすべて復元する
再保存のための受け側ディスクを挿入してください.
準備ができたらどれかキーを押してください.⏎

バックアップディスク 01 をドライブ B: に挿入してください.
準備ができたらどれかキーを押してください.⏎

*** ファイルは, 1989-9-14にバックアップされました.*** ………バックアップした日付が表示される

*** ドライブ B: からファイルを再保存します.***
ディスク番号は:  01

¥IO.SYS
¥MSDOS.SYS
¥COMMAND.COM
¥CONFIG.SYS
¥PRINT.SYS

¥MSDOS¥SYS.EXE
¥MSDOS¥DISKCOPY.EXE
¥MSDOS¥CHKDSK.EXE
¥MSDOS¥TOOL¥EDLIN.EXE

システムファイルをリストアしました.
受け側ディスクではシステムを起動できない可能性があります. ……
                        ファイル「IO.SYS」「MSDOS.SYS」
                        も復元したので警告のメッセージが表示される
A>DIR⏎ ……………ドライブA：のディスクの内容を確認する

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥
```

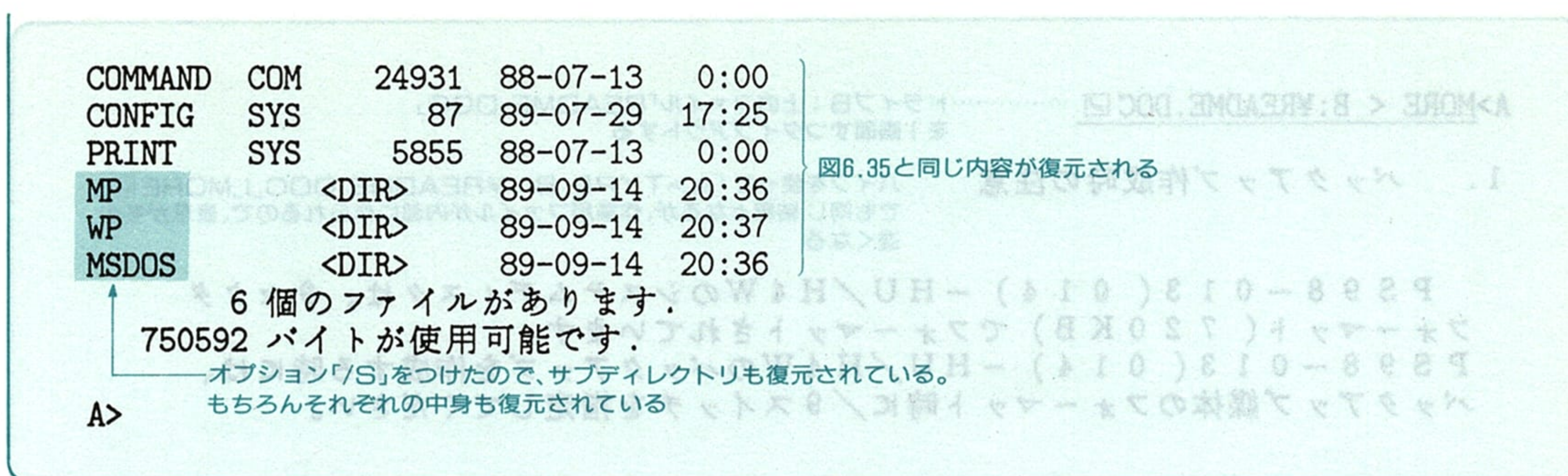
```
COMMAND  COM     24931  88-07-13   0:00
CONFIG   SYS        87  89-07-29  17:25
PRINT    SYS      5855  88-07-13   0:00
MP          <DIR>       89-09-14  20:36
WP          <DIR>       89-09-14  20:37
MSDOS       <DIR>       89-09-14  20:36
     6 個のファイルがあります.
  750592 バイトが使用可能です.

A>
```


図 6.36　バックアップした全ファイルの内容を復元する

■ スクリーン表示関係 ■

スクリーン表示をオールクリアする

［内部コマンド　　CLS］

A＞CLS⏎

スクリーン表示をオールクリアする。

スクリーン表示を連続スクロールさせず 1 画面(23 行)ごとに停止して表示する

［外部プログラム　　MORE.COM］

A＞MORE␣＜␣x：パス名¥ファイル名⏎

ドライブx：の「パス名」で指定されたディレクトリ上のテキストファイル(文字ファイル)「ファイル名」の内容を、1 画面ごと(23 行)にスクロールを停止して表示する。

A＞コマンド␣|␣MORE⏎

「コマンド」(TYPE や CHKDSK など)の表示出力を、1 画面ずつスクロールを停止して表示する。

［解説］　MORE コマンドは、前述の SORT コマンドと同じ「フィルタ」であり、リダイレクト(前者のコマンド例)、またはパイプ(後者のコマンド例)によって動作する。

```
A>MORE < B:¥README.DOC↵
```
……………ドライブB：上のファイル「README.DOC」を1画面ずつタイプアウトする

パイプを使って「A>TYPE_B：¥README.DOC_I_MORE↵」でも同じ結果となるが、作業用ファイルが内部に作られるので、表示が多少遅くなる

1．　バックアップ作成時の注意

　ＰＳ９８－０１３（０１４）－ＨＵ／Ｈ４Ｗのシステムディスクは、９セクタフォーマット（７２０ＫＢ）でフォーマットされています。
ＰＳ９８－０１３（０１４）－ＨＵ／Ｈ４Ｗのバックアップを作成する時には、バックアップ媒体のフォーマット時に／９スイッチを指定してください。

（例）

　　Ａ＞ＦＯＲＭＡＴ　Ｂ：　／９

　また、ＭＥＮＵコマンドのメニュー項目「フロッピーディスク（６４０KBﾀｲﾌﾟ）のバックアップ」を選択することにより、７２０ＫＢフロッピーディスクの初期化からコピーまで続けて行うことができます。

　メニュー項目「フロッピーディスクのバックアップ（ドライブ指定）」は、ＰＣ－９８０１ＶＸなどのノーマルモード機種では、メニュー画面の２ページ目にあります。
　メニュー画面のページを切り替えるには、ROLLUPキー（次のページを表示），
-- More -- ……………ここでスクロールがストップし、何らかのキー入力により次の1画面が表示される．CTRL＋Cで中止

ROLLDOWNキー（前のページへ戻る）を押します。

2．　バックアップ作成時のドライブ指定

　ＭＥＮＵコマンドのメニュー項目「フロッピーディスクのバックアップ」と「フロッピーディスク（６４０KBﾀｲﾌﾟ）のバックアップ」は、コピー元，コピー先のドライブがそれぞれ A:，B: 固定になっています。
A:，B:以外のドライブでバックアップを行う場合には，メニュー項目「フロッピーディスクのバックアップ（ドライブ指定）」を選択してください。

　メニュー項目「フロッピーディスクのバックアップ（ドライブ指定）」は、メニュー画面の２ページ目にあります。　メニュー画面のページを切り替えるには、ROLL-UPキー（次のページを表示），ROLLDOWNキー（前のページへ戻る）を押します。

3．固定ディスク使用時の注意

⋮

この1画面が表示されたら、再度スクロールがストップし、何らかのキー入力待ちになる

図6.37　ファイル内容の表示をページ単位に行う

プロンプト(A＞、B＞など)の形式を変更する

［内部コマンド　　PROMPT］

A＞PROMPT␣プロンプト⏎

文字列「プロンプト」で指定されたプロンプトに変更する。

A＞PROMPT⏎

プロンプトの表示を初期状態(A＞、B＞など)にもどす。

［解説］　このコマンドが実行された直後から、プロンプトの形式が指定されたプロンプトに変更される。プロンプトには、任意の文字、および次に示す記号による機能が使用できる。この設定は、コンピュータの電源をOFFにすると消滅する。

$T …… 時刻の表示
$D …… 日付の表示
$P …… カレントドライブ＋カレントディレクトリの表示
$V …… MS-DOSのバージョンNo.の表示
$N …… カレントドライブの表示
$G ……「＞」記号の表示
$L ……「＜」記号の表示
$B ……「｜」記号の表示
$$ ……「$」記号の表示
$_ …… 復帰／改行を出力
$S …… 空白(スペース)を出力
$H …… バックスペースを出力(左側の文字が消去される)
$E …… エスケープコード(1B$_{H}$)を出力(文字列と組み合わせて各種のスクリーン制御が可能になる)

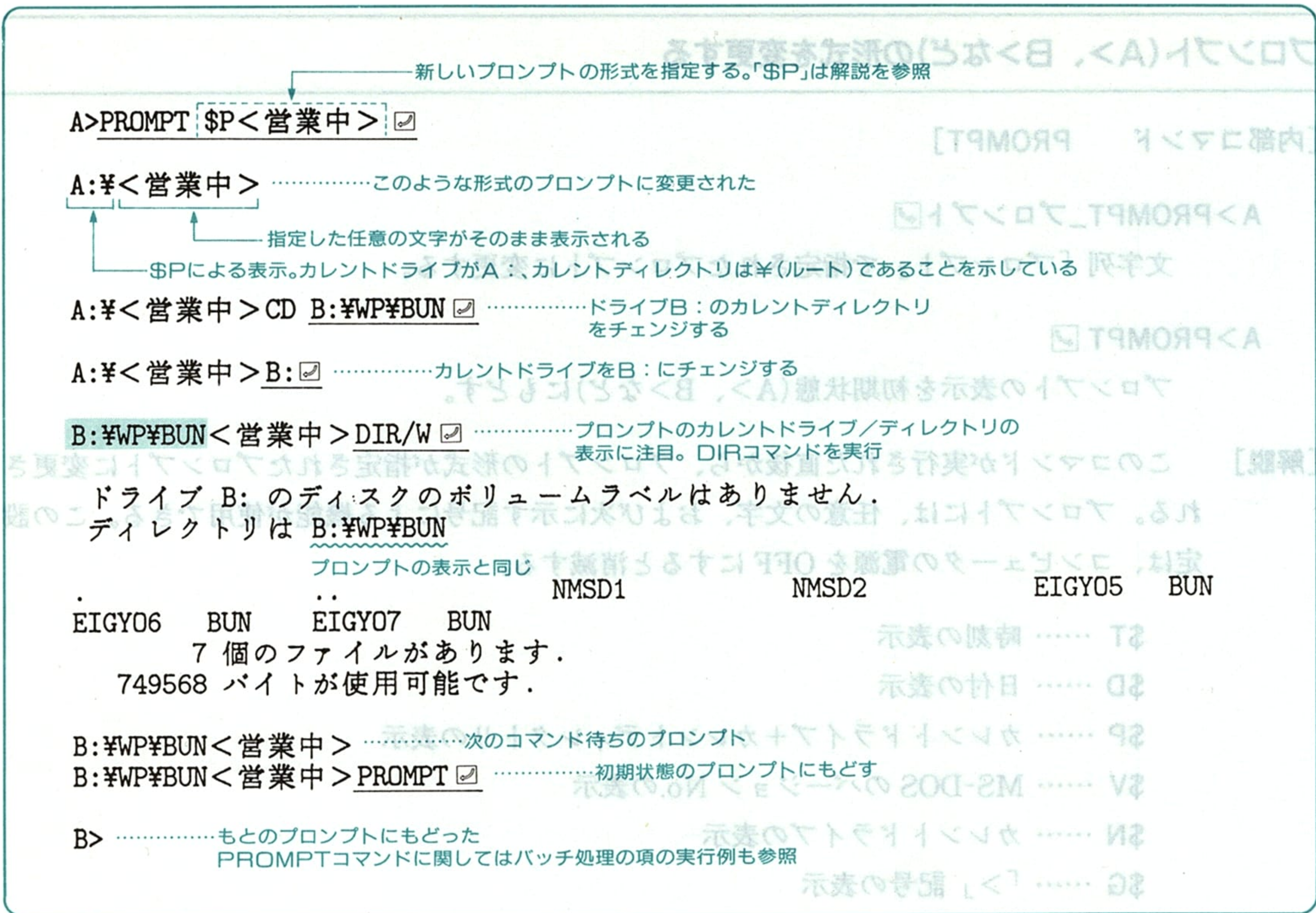
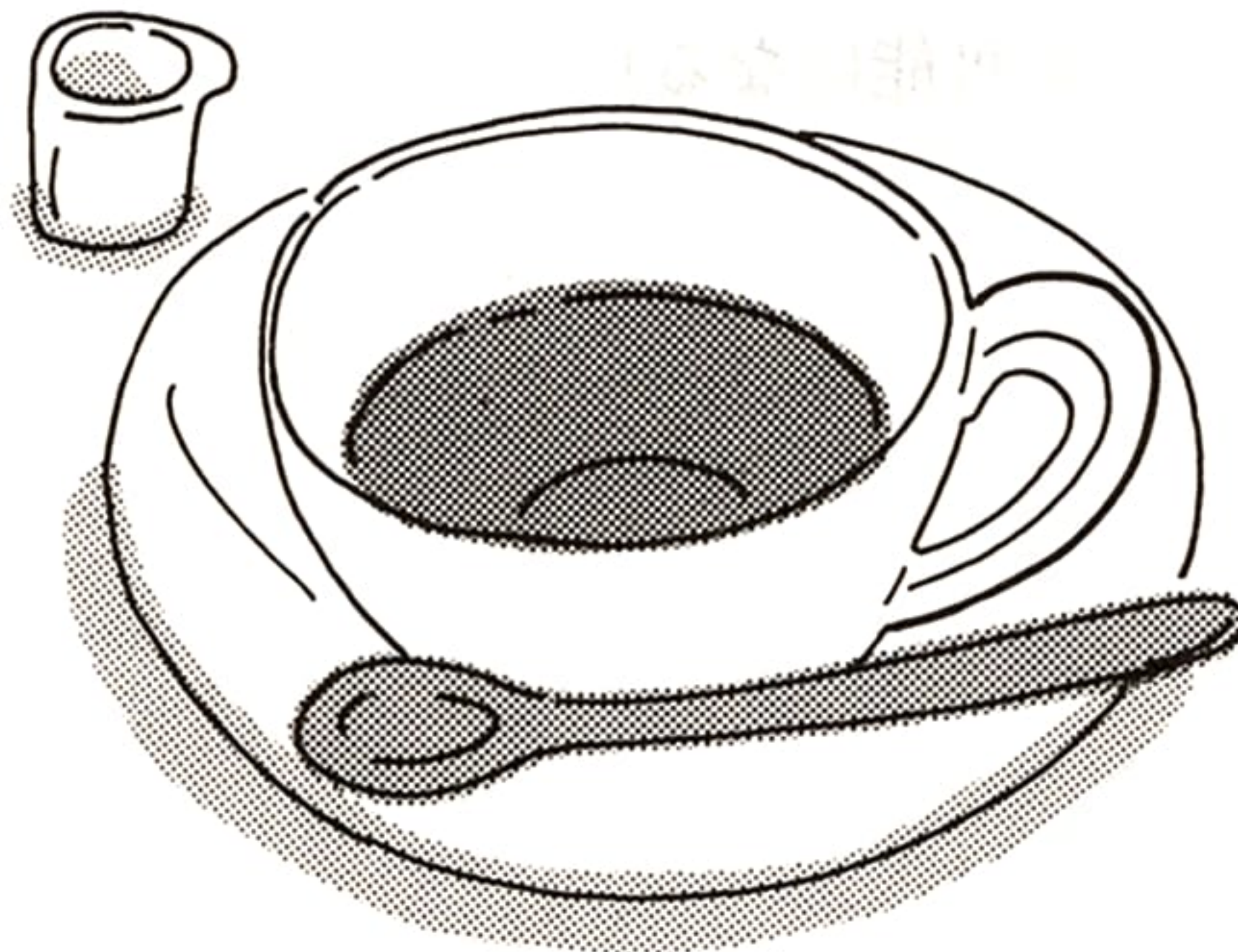
```
A>PROMPT $P<営業中>⏎
A:¥<営業中>
A:¥<営業中>CD B:¥WP¥BUN⏎
A:¥<営業中>B:⏎
B:¥WP¥BUN<営業中>DIR/W⏎
 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥WP¥BUN

.               ..              NMSD1           NMSD2           EIGYO5   BUN
EIGYO6   BUN    EIGYO7   BUN
        7 個のファイルがあります.
   749568 バイトが使用可能です.

B:¥WP¥BUN<営業中>
B:¥WP¥BUN<営業中>PROMPT⏎

B>
```


図6.38　MS-DOSのプロンプトを変更する

■ MS-DOS システムおよびコンピュータシステム関係 ■

MS-DOS のシステムファイルをコピーする

[外部プログラム　SYS.EXE]　または
[外部プログラム　FORMAT.EXE に「/S」スイッチを付ける]

A＞SYS_x：↵

カレントドライブ(この場合は A：)のディスク上の MS-DOS システムを、ドライブ x：上のディスクにコピーする。

[解説]　MS-DOS システムは、実行例にも示されているように、隠された(DIR コマンド等では表示されない)2 つのシステムファイル、「IO.SYS」、「MSDOS.SYS」と、通常のファイル「COMMAND.COM」から構成されている。

当コマンドは、カレントドライブ上(重要！)の、隠された 2 つのシステムファイルを指定ドライブにコピーする。ドライブ x：上のディスクは、フォーマットしたままの空ディスクか、あるいは、これからコピーしようとする MS-DOS システムと、まったく同じ大きさの MS-DOS システムがすでに書き込まれているディスクでなければならない。なお SYS コマンドの実行では、COMMAND.COM がコピーされないので、SYS コマンドの実行後には、COPY コマンドを使って、それをコピーしなければ MS-DOS を起動することはできない。

```
A>SYS B:↵ ……………カレントドライブ(A：)上のディスクのMS-DOSシステムを、
                  ドライブB：上のディスクにコピーする

ＳＹＳコマンド  Version 1.1
Copyright (C) 1988 NEC Corporation

システムが転送されました

A>DIR B:↵ ……………SYSコマンド実行後、ドライブB：のディレクトリを見る

 ドライブ B: のディスクのボリュームラベルは PC9800_SYS
 ディレクトリは B:¥

ファイルが見つかりません. ……………MS-DOSのシステムファイルは、DIRコマンドでは
                                  表示されない「隠された」ファイルである。
A>                                COMMAND.COMはコピーされないので、COPY
                                  コマンドでコピーしなければMS-DOSを起動すること
                                  はできない
```

図 6.39a　SYS コマンドにより MS-DOS システムをコピーする

```
A>DIR B:⏎ ……………ファイルアトリビュート変更プログラム(応用MS-DOSで作成したもの)で、「隠された
                  ファイルから通常のファイル」への変更を行った後、再びDIRコマンドを実行する

 ドライブ B: のディスクのボリュームラベルは PC9800_SYS
 ディレクトリは B:¥

IO       SYS    65536  88-07-13    0:00  隠されていた2つのMS-DOSシステムファイル。
MSDOS    SYS    29248  88-07-13    0:00  「/V」付きのCHKDSKコマンドでもこれらのファイル名
         2 個のファイルがあります.        は表示される(280ページの図6.10参照)
  1155072 バイトが使用可能です.

A>
```

図6.39b　参考までに隠されたMS-DOSシステムの2つのファイルを見る

A>FORMAT_*x*：/S⏎

ドライブ*x*：上のディスクをフォーマットし、その後にカレントドライブ上のMS-DOSシステムをコピーし、さらにCOMMAND.COMをコピーする。

［解説］　システムをコピーする前に、ドライブ*x*：上のディスクのフォーマット処理が行われるため、そのディスクの内容はすべて失われる。カレントドライブ上のMS-DOSの隠された2つのシステムファイルと、COMMAND.COMがコピーされ、完全なシステムディスクが作成される。FORMATコマンドについては、272ページの同コマンドを参照。

```
A>FORMAT B:/S⏎ ……………ドライブB：上のディスクをフォーマット処理し、かつカレントドライブ(A：)上の
Format Version 4.10     MS-DOSシステムをコピーする。カレントドライブ(A：)上のディスクには
                        COMMAND.COMが含まれていること

新しいディスクをドライブ B: に挿入し
どれかキーを押してください x

ディスクのタイプは   1 : 640(KB)  2 : 1(MB)   =  2⏎

目的のディスクは 1MB FD です

フォーマットが終了しました
0  10  20  30  40  50  60  70  80  90  100
██████████████████████████████████████████(%)
- - - - - - - - - - - - - - - - - - - - - - ←── この間にシステムがコピーされる
システムを転送しました

   1250304 バイト  全ディスク容量
    120832 バイト  システム領域
   1129472 バイト  使用可能ディスク容量
```

```
別のディスクをフォーマットしますか(Y/N) N⏎
A>DIR B:⏎ ……………システムがコピーされた後のドライブB：上のディレクトリを確認する

 ドライブ B: のディスクのボリュームラベルはありません.
 ディレクトリは B:¥

IO       SYS    65536  88-07-13    0:00 ｝この2つのファイルは、図6.39bと同様にファイルアトリ
MSDOS    SYS    29248  88-07-13    0:00 ｝ビュートを変更して、通常ファイルにしたもの。
                                          そのままの状態ではこれらのファイルは表示されない
COMMAND  COM    24931  88-07-13    0:00 ……………COMMAND.COMもコピーされている
          3 個のファイルがあります.
  1129472 バイトが使用可能です.          MS-DOSは、この3つのファイルで
                                          完全なシステムが構成される
A>
```

図 6.40　FORMAT コマンドにより MS-DOS システムをコピーする

コンピュータ内蔵のカレンダーの表示／修正

［内部コマンド　　DATE］

A＞DATE⏎

マシンに内蔵のカレンダーの年月日を表示し、必要があれば修正する。実行例は、次の項目で示す。

コンピュータ内蔵の時計の表示／修正

［内部コマンド　　TIME］

A＞TIME⏎

マシン内蔵の時計の時分秒を表示し、必要があれば修正する。

［解説］　実日付や実時刻をコンピュータが保持していることには大きな意味がある。ファイルを作成したり、更新したりすると、そのファイルにはコンピュータが保持している日時が付けられる(DIR コマンドの表示でおなじみのもの)。これを「タイムスタンプ機能」と呼ぶが、各ファイルにこのスタンプが押されていることにより、新旧ファイルの比較が可能になる。このスタンプ機能は、各種のコマンドやアプリケーションプログラムで利用され、実務コンピュータには不可欠のものである。

```
A>DATE⏎ ……………日付を修正したい
現在の日付は 1989-08-23 (水) です.
日付を入力してください: 2001-4-1⏎ …………上の行に表示されているものと同じ形式で入力する

A>TIME⏎ ……………時刻を修正したい
現在の時刻は 21:09:13.00 です.
時刻を入力してください: 12:00:00⏎ …………上の行に表示されているものと同じ形式で入力する

A>DATE⏎ ……………日付を確認したい
現在の日付は 2001-04-01 (日) です. …………変更されている
日付を入力してください:⏎ ……………修正しない場合は⏎のみを入力

A>TIME⏎ ……………時刻を確認したい
現在の時刻は 12:00:10.00 です. ……………変更されている
時刻を入力してください:⏎ ……………同上

A>
```

図 6.41　マシン内蔵のカレンダー時計を修正する

キャラクタ系のデバイスドライバを MS-DOS システムに組み込む

［<3.*x*>外部コマンド　　ADDDRV］

A＞ADDDRV_*x*：パス名¥ファイル名

ドライブ *x*：の「パス名」で指定されたディレクトリ上のファイル「ファイル名」に記述された内容のキャラクタ系デバイスドライバを MS-DOS システムに組み込む。

［解説］　ADDDRV コマンドは、DELDRV コマンドと組み合わせて、日本語フロントエンドプロセッサやマウスドライバなどのキャラクタ系デバイスドライバを切り替える場合に利用する(RAM ディスクドライバやディスクキャッシュドライバなどのブロック系デバイスドライバは組み込むことはできない)。また、すでに ADDDRV コマンドで組み込まれたデバイスドライバが存在する場合は、次項の DELDRV コマンドでそれらを削除したあとでなければ、新たなデバイスドライバを組み込むことはできない。

目的のデバイスドライバを記述しておくファイルは、CONFIG.SYS ファイルでデバイスドライバを組み込む際に記述する「DEVICE＝xxxxx」と同じ形式で作成すればよい。なお、ADDDRV コマンドおよび次項の DELDRV コマンドについては、208 ページの 4.5 章でくわしく解説している。

```
A>TYPE ATOK.DEV⏎ ······ATOK6を組み込むための定義ファイル「ATOK.DEV」の内容を見る。
                       これはエディタなどであらかじめ作成しておく
DEVICE = A:¥ATOK6A.SYS /T=1 }
DEVICE = A:¥ATOK6B.SYS      } CONFIG.SYSと同じように記述したもの

A>ADDDRV ATOK.DEV⏎ ······定義ファイル「ATOK.DEV」を指定してATOK6を組み込む
Microsoft (R) Character Device driver install and deinstall  Version 1.01 } ADDDRV
Copyright (C) Microsoft Corp 1987. All rights reserved.                   } コマンドの
                                                                            メッセージ

自由文変換システム　ＡＴＯＫ６ ver 1.2 } ATOK6の組み込みメッセージ。CONFIG.SYSで
Copyright 1986,87 (株)ジャストシステム } 組み込まれた場合と同じように表示される

A> ······CTRL+XFER キーでATOK6をONにする
         ATOK6による各種漢字変換の処理が行える
_        CTRL+XFER キーでATOK6をOFFにする                        連ローマ字漢字

A>TYPE VJEB.DEV⏎ ······VJE-βを組み込むための定義ファイル「VJEB.DEV」の内容を見る
DEVICE = VJEB.DRV /hb /m1 /t0 /l /j /k- /k, /k. /k[ /k/ /d4

A>ADDDRV VJEB.DEV⏎ ······定義ファイル「VJEB.DEV」を指定して、
                         ATOK6をVJE-βに切り替える
Microsoft (R) Character Device driver install and deinstall  Version 1.01
Copyright (C) Microsoft Corp 1987. All rights reserved.

ＡＤＤＤＲＶコマンドで組み込まれたデバイスドライバがあります．······ATOK6がすでに
                                                                  ADDDRVコマ
A>DELDRV⏎ ······DELDRVコマンドで組み込んだATOK6を削除する         ンドで組み込まれ
                                                                  ているので、VJE-β
A> ······CTRL+XFER キーを押す                                      を組み込むことが
                                                                  できない

  〔 16進〕······ATOK6が取り除かれたので、かなー漢字変換はできない
          CTRL+XFER キーを押す
A>ADDDRV VJEB.DEV⏎ ······再びVJE-βを組み込む
Microsoft (R) Character Device driver install and deinstall  Version 1.01
Copyright (C) Microsoft Corp 1987. All rights reserved.

日本語入力フロント・プロセッサ ＶＪＥ－β 第２．００版 } VJE-βの組み込みメッセージ。
Copyright (C) 1986-88 VACS Corp. / ASCII Corp.         } 今度はうまく組み込まれた。
                                                         ATOK6がVJE-βに切り替わった！

A> ······CTRL+XFER キーでVJE-βをONにする

         VJE-βによる各種漢字変換の処理が行える
かな カナ 全角 半角 ｺｰﾄﾞ 10確定 ^3ﾛｰﾏ字 ^10その他                 Rかな全J*

A>
```

図6.42　日本語入力システム(FEP)を切り替える

ADDDRVコマンドで組み込んだデバイスドライバをMS-DOSシステムから削除する

［<3.*x*>外部コマンド　　DELDRV］

A>DELDRV

ADDDRVで組み込まれたキャラクタ系デバイスドライバをMS-DOSシステムから削除する。ADDDRVで組み込まれたデバイスドライバが存在しない場合は、エラーメッセージを表示する。

［解説］　DELDRVコマンドは、ADDDRVコマンドで組み込んだデバイスドライバのみを削除する（CONFIG.SYSファイルで組み込まれたデバイスドライバは削除できない）。つまり、デバイスドライバをリセットボタンを押さずに切り替えたい場合は、CONFIG.SYSファイルに登録せず、ADDDRVコマンドで組み込む必要がある。なお、DELDRVコマンドおよび前項のADDDRVコマンドについては、208ページの4.5章でくわしく解説している。DELDRVコマンドの実行例は、前項ADDDRVコマンドを参照。

■ バッチ処理関係 ■

［すべて内部コマンド］

MS-DOSの起動と同時にMS-DOSのコマンドやユーザープログラムを実行する

［方法］　実行させたいコマンドやプログラム名を、自動スタート・バッチファイル「AUTOEXEC.BAT」に記述し、起動ディスクのルートディレクトリに置く。

```
A>TYPE AUTOEXEC.BAT ⏎    …タイプアウトで示すような内容のファイルを、ファイル名「AUTOEXEC.BAT」
                          として作成する（COPYコマンドまたはエディタを使う）
PROMPT ($P)_  ―スペース …プロンプトの変更（プロンプト変更の項参照）
CLS       ……画面のオールクリア
VER       ……MS-DOSのバージョンNo.の表示
DIR/W     ……DIRコマンドの実行

A>        ここでの例は内部コマンドばかりであるが、もちろん外部プログラム等を指定してもよい
```

⬇

[リセットボタン]を押してMS-DOSが起動すると、AUTOEXEC.BATファイルの内容が次々と自動的に実行されていく

プロンプトが変更され、その後CLSコマンドが実行され、これまでの画面はオールクリアされた

```
(A:¥) VER ……VERコマンドの実行

MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30            これらのコマンドラインが表示されて
                               いることに注目しておく

(A:¥) DIR/W ……DIRコマンドの実行

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

COMMAND  COM    ASSIGN   EXE    ATTRIB   EXE    BACKUP   EXE    CHKDSK   EXE
CHKFIL   EXE    CUSTOM   EXE    DISKCOPY EXE    DUMP     EXE    EDLIN    EXE
FC       EXE    FORMAT   EXE    KEY      EXE    REPLACE  EXE    RESTORE  EXE
SETUP2   EXE    SPEED    EXE    SWITCH   EXE    SYMDEB   EXE    SYS      EXE
TREE     EXE    USKCGM   EXE    XCOPY    EXE    CONFIG   SYS    EMSDRIVE SYS
GRAPH    SYS    MOUSE    SYS    PRINT    SYS    RAMDISK  SYS    RSDRV    SYS
README   DOC    FIND     EXE    DIRNAME         DIRSIZE         AUTOEXEC BAT
       35 個のファイルがあります.
   473088 バイトが使用可能です.

(A:¥)
(A:¥) ……MS-DOSが起動して、自動スタートによる一連の実行が終わり、入力待ちとなっている
```

⬇

```
(A:¥) B:⏎
                  変更されたプロンプトは、カレントドライブおよび
(B:¥) ⏎           カレントディレクトリを常に表示する
(B:¥) CD ¥MP⏎

(B:¥MP) ……入力待ち
```

図 6.43　MS-DOS の起動時に、コマンドやプログラムを自動実行させる

バッチファイルの実行時に、コマンドラインを表示しない

［ECHO］（バッチ処理コマンド）

ECHO_OFF

バッチファイル内の、この行以降のプロンプトやコマンドラインを表示しない。

ECHO_ON

この行以降、コマンドラインの表示を再開する。また、バッチ処理が終わると自動的にONとなる。バージョン3.*x*では、「ECHO」でECHOのON／OFFの状態が表示される。

```
A>TYPE AUTOEXEC.BAT        ……タイプアウトして示すように、先ほどのAUTOEXEC.BATファイルに
                              ECHOコマンドを追加した
ECHO OFF    ……この行を追加した。プロンプトおよびコマンドラインの表示をOFFにする
PROMPT ($P)
CLS
VER
DIR/W
   ……ECHO ONがなくても、バッチファイルの終了と同時に自動的にONとなる
A>
前項と同様にリセットボタンを押してMS-DOSを起動する。今回はこのような実行画面となった
A>ECHO OFF
   ……プロンプトとVERコマンドの表示はされない
MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30
   ……DIRコマンドの表示がされていない
 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

COMMAND  COM    ASSIGN   EXE    ATTRIB   EXE    BACKUP   EXE    CHKDSK   EXE
CHKFIL   EXE    CUSTOM   EXE    DISKCOPY EXE    DUMP     EXE    EDLIN    EXE
FC       EXE    FORMAT   EXE    KEY      EXE    REPLACE  EXE    RESTORE  EXE
SETUP2   EXE    SPEED    EXE    SWITCH   EXE    SYMDEB   EXE    SYS      EXE
TREE     EXE    USKCGM   EXE    XCOPY    EXE    CONFIG   SYS    EMSDRIVE SYS
GRAPH    SYS    MOUSE    SYS    PRINT    SYS    RAMDISK  SYS    RSDRV    SYS
README   DOC    FIND     EXE    DIRNAME         DIRSIZE         AUTOEXEC BAK
AUTOEXEC BAT
         36 個のファイルがあります.
     472064 バイトが使用可能です.

(A:¥)   ……バッチファイルの実行が終了すると、ECHOは自動的にONとなる
```

図6.44　ECHOコマンドによりコマンドラインの表示をON／OFFする

いくつかのコマンドを 1 つのバッチファイルで一括自動実行させる

［方法］　実行させたいコマンドやプログラムを、実行順にファイルタイプ(拡張子)が「.BAT」の任意のバッチファイル「xxxxxxxx.BAT」に登録し、そのファイル名をキー入力⏎することにより実行される。

```
                適当な名前を付ける
(A:¥) REN AUTOEXEC.BAT MANUEXEC.BAT⏎ ……………前項の自動スタート・バッチファイルから、通常の
                                              バッチファイルにファイル名を変更する
(A:¥) ……………これでMS-DOS起動時の自動スタート機能は失われた

(リセットボタンを押す)
NEC PC-9800 Series Personal Computer

ﾏｲｸﾛｿﾌﾄ MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30
Copyright 1981,88 Microsoft Corp. / NEC Corporation

    プリンタが使用可能です

    ＲＳ－２３２Ｃインターフェイスが使用可能です

Command ﾊﾞｰｼﾞｮﾝ 3.30
現在の日付は 1989-08-09 (水) です.
日付を入力してください:⏎
現在の時刻は 20:04:04.00 です.    自動スタート・バッチファイルは存在しないので、MS-DOSが
時刻を入力してください:⏎          起動し、入力待ちとなる

A>MANUEXEC⏎ ……………バッチファイルをキー入力により実行する

A>ECHO OFF
```

⬇

```
MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

COMMAND  COM   ASSIGN   EXE   ATTRIB   EXE   BACKUP   EXE   CHKDSK   EXE
CHKFIL   EXE   CUSTOM   EXE   DISKCOPY EXE   DUMP     EXE   EDLIN    EXE
FC       EXE   FORMAT   EXE   KEY      EXE   REPLACE  EXE   RESTORE  EXE
SETUP2   EXE   SPEED    EXE   SWITCH   EXE   SYMDEB   EXE   SYS      EXE
TREE     EXE   USKCGM   EXE   XCOPY    EXE   CONFIG   SYS   EMSDRIVE SYS
GRAPH    SYS   MOUSE    SYS   PRINT    SYS   RAMDISK  SYS   RSDRV    SYS
README   DOC   FIND     EXE   DIRNAME        DIRSIZE        AUTOEXEC BAK
AUTOEXEC BAT
        36 個のファイルがあります.
    472064 バイトが使用可能です.

(A:¥)
```

図6.44と同様に、一連のコマンドが実行された
なお、「AUTOEXE.BAT」のままでも、「AUTOEXEC⏎」とすれば、通常のバッチファイルと同様に任意のときに実行することができる。

図6.45　いくつかのコマンドをバッチファイルで一括自動実行する

バッチファイル実行中に任意のメッセージを表示する

［ECHO および REM］　（バッチ処理コマンド）

ECHO_メッセージ

ECHOがONの場合、「ECHO」を含めて「メッセージ」が表示される。ECHOがOFFの場合、「メッセージ」のみが表示される。

REM_メッセージ

ECHOがONの場合、「REM」を含めて「メッセージ」が表示される。ECHOがOFFの場合は、何も表示されない。

```
A>TYPE MANUEXEC.BAT⏎ ……………ECHOおよびREMコマンドによりメッセージ行を
                              追加したバッチファイルをタイプアウトする
ECHO OFF
PROMPT ($P)
CLS
ECHO こんにちは! ……………①
VER
REM どうぞよろしく ……………②
DIR/W
ECHO ON ……………ここでECHOをONにしていることに注目
REM どうぞよろしく ……………③

A>MANUEXEC⏎ ……………このバッチファイルを実行する

A>ECHO OFF
```

⬇

```
こんにちは! ……………①による表示

MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30
……………②による表示はECHOがOFFのため表示されない
 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

COMMAND  COM    ASSIGN   EXE    ATTRIB   EXE    BACKUP   EXE    CHKDSK   EXE
CHKFIL   EXE    CUSTOM   EXE    DISKCOPY EXE    DUMP     EXE    EDLIN    EXE
FC       EXE    FORMAT   EXE    KEY      EXE    REPLACE  EXE    RESTORE  EXE
SETUP2   EXE    SPEED    EXE    SWITCH   EXE    SYMDEB   EXE    SYS      EXE
TREE     EXE    USKCGM   EXE    XCOPY    EXE    ADDDRV   EXE    EMSDRIVE SYS
GRAPH    SYS    MOUSE    SYS    PRINT    SYS    RAMDISK  SYS    RSDRV    SYS
README   DOC    FIND     EXE    VJE      DEV    MANUEXEC BAT    CONFIG   SYS
        35 個のファイルがあります.
    455680 バイトが使用可能です.
……………この時点でECHOがONになる
(A:¥) REM どうぞよろしく ……………③による表示

(A:¥)
(A:¥)
```

図 6.46 バッチファイルの実行中に任意のメッセージを表示させる

バッチファイルの実行を一時ストップする

[PAUSE]　(バッチ処理コマンド)

PAUSE

ECHOがONの場合、「PAUSE　準備ができたらどれかキーを押してください」と表示され、バッチ処理が一時ポーズ状態(ストップ状態)になる。ECHOがOFFの場合、「PAUSE」の部分が表示されずに、それ以降のメッセージのみ表示され、ポーズする。

[解説]　PAUSEコマンドは、バッチファイルの実行途中、その位置で実行を一時ストップし、「メッセージ」を表示する。ユーザーはこの間に、たとえばディスクの交換などを行うことができる。バッチ処理は、何らかのキー入力で再開され、CTRL＋Cを入力した場合は、以降のバッチ処理を中止することができる。

```
A>TYPE MANUEXEC.BAT
ECHO OFF
PROMPT ($P)
CLS
ECHO こんにちは!
PAUSE
VER
DIR/W

A>MANUEXEC

A>ECHO OFF
こんにちは!
準備ができたらどれかキーを押してください.

MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

COMMAND  COM    ASSIGN  EXE    ATTRIB   EXE    BACKUP   EXE    CHKDSK   EXE
CHKFIL          DISKCOPY EXE   DUMP     EXE    EDLIN    EXE
                               REPLACE  EXE    RESTORE  EXE
README   DOC    FIND    EXE    VJE
       35 個のファイルがあります.
   455680 バイトが使用可能です.

(A:¥)
```

PAUSEコマンドを挿入したバッチファイルをタイプアウトする
ECHOをOFFにしていることに注目
PAUSEコマンドをここに挿入
このバッチファイルを実行する
PAUSEコマンドによる表示
ここでポーズ状態となる。何らかのキー入力で以降のコマンドが実行される

図6.47　バッチファイルの実行を一時ストップさせる

バッチファイルの実行の流れを変える

［GOTO］（バッチ処理コマンド）

GOTO_ラベル

このコマンド以降、実行は「：ラベル」の行に移る。

［解説］　BASICのGOTO文と同じ働きをし、指定したラベルの行にジャンプする。ラベルには「：」を前置する。次項の条件実行のIFコマンドと組み合わせると、いろいろな条件分岐を行うことができる。

```
A>TYPE MANUEXEC.BAT ⏎ ……GOTOコマンドとラベルを挿入したバッチファイルをタイプアウトする
ECHO OFF
PROMPT ($P)
CLS
:LOOP ……ラベル
ECHO こんにちは!
PAUSE
VER
DIR/W
GOTO LOOP ……ラベル「LOOP」へジャンプする

A>MANUEXEC ⏎ ……このバッチファイルを実行する

A>ECHO OFF
こんにちは!
準備ができたらどれかキーを押してください. ……ポーズする

MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥

COMMAND  COM    ASSIGN  EXE    ATTRIB  EXE    BACKUP   EXE    CHKDSK  EXE
               ∫
README   DOC    FIND    EXE    VJE     DEV    MANUEXEC BAT    CONFIG  SYS
        35 個のファイルがあります.
    455680 バイトが使用可能です.
こんにちは!
準備ができたらどれかキーを押してください. ……ポーズする。以降同様の繰り返しとなる

MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30

 ドライブ A: のディスクのボリュームラベルはありません.
 ディレクトリは A:¥
               ∫
```

（実行の流れ）

ポーズ状態において、CTRL＋Cの入力により、バッチ処理を中止できる

```
              ∫
GRAPH    SYS    MOUSE    SYS    PRINT    SYS    RAMDISK  SYS    RSDRV    SYS
README   DOC    FIND     EXE    VJE      DEV    MANUEXEC BAT    CONFIG   SYS
        35 個のファイルがあります.
    455680 バイトが使用可能です.
こんにちは!
準備ができたらどれかキーを押してください. ^C ……………CTRL+C を入力する

バッチ処理を中止しますか <Y/N>? Y
(A:¥) ……………バッチ処理が中止され、MS-DOSのプロンプトにもどった
```

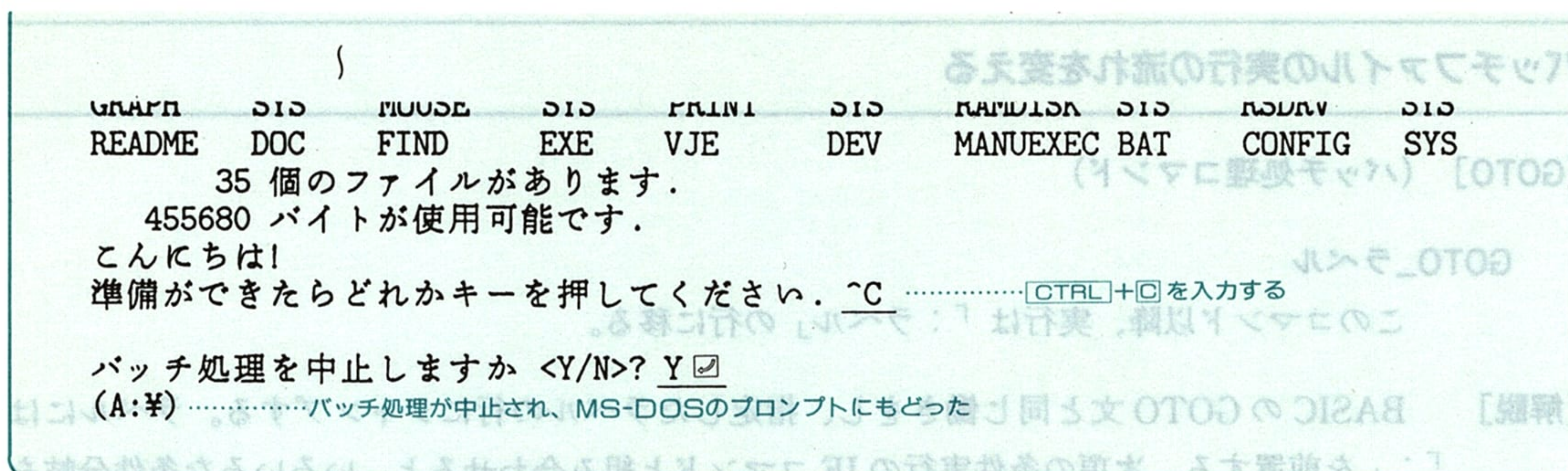



図 6.48　GOTO コマンドによりバッチファイルの実行を分岐させる

ディスク上に指定したファイルが存在する場合／しない場合、指定コマンドを実行する

［IF　EXIST］　(バッチ処理コマンド)

IF_EXIST_*x*：パス名¥ファイル名_コマンド

もし、ドライブ*x*：の「パス名」で指定されるディレクトリ上に、ファイル「ファイル名」が存在する場合、「コマンド」を実行する。

IF_NOT_EXIST_*x*：パス名¥ファイル名_コマンド

もし、ドライブ*x*：の「パス名」で指定されるディレクトリ上に、ファイル「ファイル名」が存在しない場合、「コマンド」を実行する。

［解説］　前項の GOTO 文と組み合わせると複雑な条件分岐が可能になる。

ドライブB：上のSORTプログラム「SORT.EXE」を使って、ドライブAのディレクトリ表示をソートする。もしドライブB：上にSORTプログラムが存在しない場合は、警告メッセージを表示する。このような内容のバッチファイルを作成する

```
A>TYPE MANUEXEC.BAT ……………そのバッチファイルをタイプアウトする
ECHO OFF
PROMPT ($P)     今までと同じもの
CLS
VER
IF NOT EXIST B:SORT.EXE ECHO ドライブB: に「SORT.EXE」がありません. 正しいディ
スクをセットしてください.
IF NOT EXIST B:SORT.EXE PAUSE
DIR *.EXE | B:SORT
     (IF文 / 条件実行するコマンド)

A>MANUEXEC ……………ドライブB：上に「SORT.EXE」がない場合の実行例
```

```
A>ECHO OFF

MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30
ドライブB: に「SORT.EXE」がありません. 正しいディスクをセットしてください.
準備ができたらどれかキーを押してください.

       24 個のファイルがあります.
    453632 バイトが使用可能です.
 ディレクトリは A:¥
 ドライブ A: のディスクのボリュームラベルはありません.
ADDDRV    EXE     18384  88-07-13     0:00
ASSIGN    EXE     26450  88-07-13     0:00
ATTRIB    EXE      9154  88-07-13     0:00
TREE      EXE      9608  88-07-13     0:00
USKCGM    EXE     22444  88-07-13     0:00
XCOPY     EXE      5768  88-07-13     0:00

(A:¥)
```


図 6.49a 条件分岐のバッチファイルを実行する

```
A>MANUEXEC ⏎
```



```
A>ECHO OFF

MS-DOS ﾊﾞｰｼﾞｮﾝ 3.30

       24 個のファイルがあります.
    453632 バイトが使用可能です.
 ディレクトリは A:¥
 ドライブ A: のディスクのボリュームラベルはありません.
ADDDRV    EXE     18384  88-07-13     0:00
ASSIGN    EXE     26450  88-07-13     0:00
ATTRIB    EXE      9154  88-07-13     0:00
BACKUP    EXE     25218  88-07-13     0:00
SYS       EXE     25480  88-07-13     0:00
TREE      EXE      9608  88-07-13     0:00
USKCGM    EXE     22444  88-07-13     0:00
XCOPY     EXE      5768  88-07-13     0:00

(A:¥)
```


図 6.49b 条件分岐のバッチファイルを実行する（上と逆の条件の場合）

APPENDIX

ここに、2章で解説したアスキーコード・キャラクタ表と、シフトJIS、およびJIS漢字コード表の一部を示しておきます。JIS-C6226で定義されていない漢字コードの未定義エリア(たとえばJISコードの2780_H〜$2FFF_H$など)には、各社独自の文字や記号が割り当てられている機種もあります。

■アスキーコード・キャラクタ表

上位4ビット→　下位4ビット↓

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		DE		0	@	P		p				ー	タ	ミ		╳
1	SH	D1	!	1	A	Q	a	q			。	ア	チ	ム		円
2	SX	D2	"	2	B	R	b	r			「	イ	ツ	メ		年
3	EX	D3	#	3	C	S	c	s			」	ウ	テ	モ		月
4	ET	D4	$	4	D	T	d	t			、	エ	ト	ヤ		日
5	EQ	NK	%	5	E	U	e	u			・	オ	ナ	ユ		時
6	AK	SN	&	6	F	V	f	v			ヲ	カ	ニ	ヨ		分
7	BL	EB	'	7	G	W	g	w			ァ	キ	ヌ	ラ		秒
8	BS	CN	(	8	H	X	h	x			ィ	ク	ネ	リ	♠	
9	HT	EM	)	9	I	Y	i	y			ゥ	ケ	ノ	ル	♥	
A	LF	SB	*	:	J	Z	j	z			ェ	コ	ハ	レ	♦	
B	HM	EC	+	;	K	[	k	{			ォ	サ	ヒ	ロ	♣	
C	CL	→	,	<	L	¥	l	¦			ャ	シ	フ	ワ	●	
D	CR	←	−	=	M	]	m	}			ュ	ス	ヘ	ン	○	
E	SO	↑	.	>	N	∧	n	~			ョ	セ	ホ	゛		
F	SI	↓	／	?	O	_	o				ッ	ソ	マ	゜		

この部分は文字を表すエリアではなく、各種のコントロールコードに割り当てられている。

「JIS-C6220」で未定義のエリアであり、各社独自のグラフィックキャラクタ等に使われている。シフトJIS漢字コードでは、このエリアのものを第1バイト目として使用している。

■JIS漢字コード表のみかた

（シフトJIS：$889F_{H}$　JIS：3021_{H}）

（シフトJIS：$824F_{H}$　JIS：2330_{H}）

（シフトJIS：$88A4_{H}$　JIS：3026_{H}）

（シフトJIS：8260_{H}　JIS：2341_{H}）

（シフトJIS：$88AD_{H}$　JIS：$302F_{H}$）

（シフトJIS：$82A0_{H}$　JIS：2422_{H}）

	シフトJISコード	JISコード	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
記号	813F	2120			、	。	，	．	・	：	；	？	！	゛	゜	´	｀	¨
	814F	2130	＾	￣	＿	ヽ	ヾ	ゝ	ゞ	〃	仝	々	〆	〇	ー	―	‐	／
	815F	2140	＼	～	∥	｜	…	‥	‘	’	“	”	（	）	〔	〕	［	］
	816F	2150	｛	｝	〈	〉	《	》	「	」	『	』	【	】	＋	－	±	×
	8180	2160	÷	＝	≠	＜	＞	≦	≧	∞	∴	♂	♀	°	′	″	℃	￥
	8190	2170	＄	¢	£	％	＃	＆	＊	＠	§	☆	★	○	●	◎	◇	
	819E	2220		◆	□	■	△	▲	▽	▼	※	〒	→	←	↑	↓	＝	
英・数字	824F	2330	0	1	2	3	4	5	6	7	8	9						
	825F	2340		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	826F	2350	P	Q	R	S	T	U	V	W	X	Y	Z					
	8280	2360		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
	8290	2370	p	q	r	s	t	u	v	w	x	y	z					
ひらがな	829E	2420		ぁ	あ	ぃ	い	ぅ	う	ぇ	え	ぉ	お	か	が	き	ぎ	く
	82AE	2430	ぐ	け	げ	こ	ご	さ	ざ	し	じ	す	ず	せ	ぜ	そ	ぞ	た
	82BE	2440	だ	ち	ぢ	っ	つ	づ	て	で	と	ど	な	に	ぬ	ね	の	は
	82CE	2450	ば	ぱ	ひ	び	ぴ	ふ	ぶ	ぷ	へ	べ	ぺ	ほ	ぼ	ぽ	ま	み
	82DE	2460	む	め	も	ゃ	や	ゅ	ゆ	ょ	よ	ら	り	る	れ	ろ	ゎ	わ
	82EE	2470	ゐ	ゑ	を	ん												
カタカナ	833F	2520		ァ	ア	ィ	イ	ゥ	ウ	ェ	エ	ォ	オ	カ	ガ	キ	ギ	ク
	834F	2530	グ	ケ	ゲ	コ	ゴ	サ	ザ	シ	ジ	ス	ズ	セ	ゼ	ソ	ゾ	タ
	835F	2540	ダ	チ	ヂ	ッ	ツ	ヅ	テ	デ	ト	ド	ナ	ニ	ヌ	ネ	ノ	ハ
	836F	2550	バ	パ	ヒ	ビ	ピ	フ	ブ	プ	ヘ	ベ	ペ	ホ	ボ	ポ	マ	ミ
	8380	2560	ム	メ	モ	ャ	ヤ	ュ	ユ	ョ	ヨ	ラ	リ	ル	レ	ロ	ヮ	ワ
	8390	2570	ヰ	ヱ	ヲ	ン	ヴ	ヵ	ヶ									
ギリシア文字	839E	2620		Α	Β	Γ	Δ	Ε	Ζ	Η	Θ	Ι	Κ	Λ	Μ	Ν	Ξ	Ο
	83AE	2630	Π	Ρ	Σ	Τ	Υ	Φ	Χ	Ψ	Ω							
	83BE	2640		α	β	γ	δ	ε	ζ	η	θ	ι	κ	λ	μ	ν	ξ	ο
	83CE	2650	π	ρ	σ	τ	υ	φ	χ	ψ	ω							
ロシア文字	843F	2720		А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н
	844F	2730	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э
	845F	2740	Ю	Я														
	846F	2750		а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н
	8480	2760	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э
	8490	2770	ю	я														

	シフトJISコード	JISコード	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
あ	889E	3020		亜	唖	娃	阿	哀	愛	挨	姶	逢	葵	茜	穐	悪	握	渥
	88AE	3030	旭	葦	芦	鯵	梓	圧	斡	扱	宛	姐	虻	飴	絢	綾	鮎	或
	88BE	3040	粟	袷	安	庵	按	暗	案	闇	鞍	杏						
い	88BE	3040											以	伊	位	依	偉	囲
	88CE	3050	夷	委	威	尉	惟	意	慰	易	椅	為	畏	異	移	維	緯	胃
	88DE	3060	萎	衣	謂	違	遺	医	井	亥	域	育	郁	磯	一	壱	溢	逸
	88EE	3070	稲	茨	芋	鰯	允	印	咽	員	因	姻	引	飲	淫	胤	蔭	
	893F	3120		院	陰	隠	韻	吋										
う	893F	3120							右	宇	烏	羽	迂	雨	卯	鵜	窺	丑
	894F	3130	碓	臼	渦	嘘	唄	欝	蔚	鰻	姥	厩	浦	瓜	閏	噂	云	運
	895F	3140	雲															
え	895F	3140		荏	餌	叡	営	嬰	影	映	曳	栄	永	泳	洩	瑛	盈	穎
	896F	3150	頴	英	衛	詠	鋭	液	疫	益	駅	悦	謁	越	閲	榎	厭	円
	8980	3160	園	堰	奄	宴	延	怨	掩	援	沿	演	炎	焔	煙	燕	猿	縁
	8990	3170	艶	苑	薗	遠	鉛	鴛	塩									
お	8990	3170								於	汚	甥	凹	央	奥	往	応	
	899E	3220		押	旺	横	欧	殴	王	翁	襖	鴬	鴎	黄	岡	沖	荻	億
	89AE	3230	屋	憶	臆	桶	牡	乙	俺	卸	恩	温	穏	音				
か	89AE	3230													下	化	仮	何
	89BE	3240	伽	価	佳	加	可	嘉	夏	嫁	家	寡	科	暇	果	架	歌	河
	89CE	3250	火	珂	禍	禾	稼	箇	花	苛	茄	荷	華	菓	蝦	課	嘩	貨
	89DE	3260	迦	過	霞	蚊	俄	峨	我	牙	画	臥	芽	蛾	賀	雅	餓	駕
	89EE	3270	介	会	解	回	塊	壊	廻	快	怪	悔	恢	懐	戒	拐	改	
	8A3F	3320		魁	晦	械	海	灰	界	皆	絵	芥	蟹	開	階	貝	凱	劾
	8A4F	3330	外	咳	害	崖	慨	概	涯	碍	蓋	街	該	鎧	骸	浬	馨	蛙
	8A5F	3340	垣	柿	蛎	鈎	劃	嚇	各	廓	拡	撹	格	核	殻	獲	確	穫
	8A6F	3350	覚	角	赫	較	郭	閣	隔	革	学	岳	楽	額	顎	掛	笠	樫
	8A80	3360	橿	梶	鰍	潟	割	喝	恰	括	活	渇	滑	葛	褐	轄	且	鰹
	8A90	3370	叶	椛	樺	鞄	株	兜	竃	蒲	釜	鎌	噛	鴨	栢	茅	萱	
	8A9E	3420		粥	刈	苅	瓦	乾	侃	冠	寒	刊	勘	勧	巻	喚	堪	姦
	8AAE	3430	完	官	寛	干	幹	患	感	慣	憾	換	敢	柑	桓	棺	款	歓
	8ABE	3440	汗	漢	澗	潅	環	甘	監	看	竿	管	簡	緩	缶	翰	肝	艦
	8ACE	3450	莞	観	諌	貫	還	鑑	間	閑	関	陥	韓	館	舘	丸	含	岸
	8ADE	3460	巌	玩	癌	眼	岩	翫	贋	雁	頑	顔	願					

（シフトJIS：$838F_{H}$　JIS：$256F_{H}$）

（シフトJIS：$82AD_{H}$　JIS：$242F_{H}$）

（シフトJIS：$8A43_{H}$　JIS：3324_{H}）

（シフトJIS：$89C4_{H}$　JIS：3246_{H}）

おわりに

いかがでしたでしょうか。ハードディスクをお使いの方は、FEPやワープロソフトを切り替える手法を実験されたかもしれませんね。旧バージョンのMS-DOSとフロッピーディスクのシステムでは考えられないこのような切り替えが、簡単かつスピーディに行えるのも、新しい環境のおかげです。私はまず、本書に一通り目を通された方には(ハードディスクをお使いの方もフロッピーディスクをお使いの方も)、1～2Mバイト程度のディスクキャッシュを利用されることをお勧めします。これは誰にでも、どのような場合にも勧められる安全で最も効果的なマシンのパワーアップ法です。

よい仕事を行うには、やはりよい環境を作り出すことが必要です。MS-DOSマシンはまだまだ快適に使うことができます。そのために本書を活用してください。そしてその快適さを実現した後、『こんなことならもっと早く装備すべきだった』と思っていただくのも本書の役目の1つなのです。

索　引

L

M

N

P

Q

R

S

T

V

X

ア

カ

サ

タ

ナ

ハ

マ

ヤ

ラ

ワ

・MS-DOS は、米国 Microsoft 社の商標です。
・一太郎、ATOK は、株式会社ジャストシステムの登録商標です。
・松、松茸は、株式会社管理工学研究所の登録商標です。
・VJE は、株式会社アスキーと株式会社バックスの登録商標です。
・MELWARE は、株式会社メルコの商品名です。
・Lotus1-2-3 は、米国 Lotus Development 社の登録商標です。
・MIFES は、メガソフト株式会社の商品名です。

編集協力：藤本　衡／中村　和志

実用 MS-DOS　改訂新版
アスキー・ラーニングシステム②実用コース

1985年7月28日　初版発行
1989年12月21日　第3版第1刷発行
1990年6月1日　第3版第4刷発行
定価1,900円(本体1,845円)

著　者　村瀬康治（むらせやすはる）
発行者　塚本慶一郎
発行所　株式会社アスキー
　〒107-24　東京都港区南青山6-11-1スリーエフ南青山ビル
振　替　東京4-161144
TEL (03)486-7111（大代表）
情報 TEL (03)498-0299（ダイヤルイン）
出版営業部 TEL (03)486-1977（ダイヤルイン）

制　作　株式会社 GARO
印　刷　株式会社 加藤文明社印刷所

編　集　　佐藤英一

ISBN 4-7561-0017-1　　**Printed in Japan**

● 11097／11153／11404／11492